1993年3月1日発行（毎月1回1日発行）第12巻3号通巻131号 昭和58年11月2日第三種郵便物認可

PERSONAL COMPUTER MAGAZINE for MZ, X1, and X68000

特集 X-BASICを学ぶ

MIRAGE System Model Stuff/MIDI音源 SC-33/JW-50
ワンチップIC工作入門/アクセラレータ(その3)

緊急速報 待望の32ビット X68030登場!

1993

SOFT BANK オー！エックス 定価600円

SHARP
目の付けどころが、
シャープでしょ。
夢の、頂きへ。
68ワールドの最高峰。
X68030
32bit PERSONAL WORKSTATION

32ビットパーソナルワークステーション

演算速度4.3倍(当社10MHz機比)/2.4倍(当社XVI比)※1、動画ウィンドウに見る新創造次元。選ばれた人だけが持つ感性によってX68030の扉はひらかれる。

X68000シリーズとして初の32ビットMPU MC68EC030を搭載して高速化を実現。

データキャッシュ、プログラムキャッシュをそれぞれ256バイト搭載したクロック周波数25MHzの高速32ビットMPUを搭載。演算速度は2倍以上(当社従来比)※1の高速化を実現しました。また数値演算プロセッサMC68882※2(25MHz)もサポート。大量の実数演算を必要とするクリエイティブワークやGUI環境の操作性など、実行速度の飛躍的な向上が図られています。

※1Dhrystn(四則演算)比。25MHz・データキャッシュオン・プログラムキャッシュオンでMC68000/10MHz時の約4.3倍、16MHz時の約2.4倍。

※2数値演算プロセッサCZ-5MP1(近日発売):本体内の専用ソケットに取りつけ可能。

65,536色表示、動画表示を実現。さらにパワーアップしたSX-WINDOWver.3.0。

X68000独自の本格的ウィンドウシステムとして定評の「SX－WINDOWver.2.0」をさらに強化した「SX－WINDOWver.3.0」を標準装備。新たに、65,536色の自然色グラフィック表示を可能とした『グラフィックウィンドウ』※を搭載。またアニメーション動画をウィンドウ上で表現でき、手軽にコンピュータアニメーションが楽しめる『CGAウィンドウ』、さらに従来のエディタのイメージを一新、高度な日本語文書作成をサポートするSX-WINDOW対応の高機能日本語マルチフォントエディタを標準装備。アウトラインフォントの展開もさらに高速化が図られています。

※SX-WINDOW上の512×512ドットのエリア内で表示可能。

GUIに対応する大容量メインメモリを搭載。

メインメモリは標準で4Mバイト、複数のアプリケーションをウィンドウ上で同時に使用するなど大量のデータ処理に対応。また本体内の増設で、I/Oスロットを使用せず最大12Mバイトまで拡張できます。拡張したメモリはすべて32ビットバスによる高速アクセスが可能、優れた拡張環境でシステムパワーアップをサポートします。

※メモリ増設には、4MB内部増設メモリボードCZ-5BE4(近日発売)、4MB増設メモリモジュールCZ-5ME4(近日発売)をご使用ください。なおCZ-5ME4はCZ-5BE4上に装着します。

X68000シリーズの高機能を継承した上で、さらに使いやすさの向上を図ったコンパチビリティ重視設計※1、すぐに使える高機能ソフトを標準装備。

●25MHzでは速すぎるアプリケーションも、従来のクロック周波数(10MHz/16MHz)で動作可能なソフトコンパチ重視設計●65,536色同時発色の自然色グラフィックス(最大表示エリア512×512ドット)、1024×1024ドットの実画面エリアを持つ高解像度表示能力(最大表示エリア768×512ドット・カラー液晶ディスプレイ使用時※2は640×480ドット)、疑似高解像度スーパーインポーズ(インターレース方式/512×512ドット・専用ディスプレイテレビ使用時)を装備した高精細度自然色グラフィックス機能。●外部MIDI音源もコントロール可能※3、ウィンドウ上で手軽にコンピュータミュージックが楽しめるMIDI音源対応デバイスドライバ搭載●ステレオ8オクターブ8重和音FM音源、ADPCM搭載●プリンタ、RS-232C、SCSI、オーディオ入出力、イメージ入力など多彩なインターフェイスを装備。●日本語変換効率や操作性を高めた日本語フロントプロセッサASKver.3.0搭載。●従来のエディタのイメージを一新したSX-WINDOW対応の高速多機能日本語マルチフォントエディタ標準装備●日本語マルチフォントエディタ中に貼り付ける絵やグラフなどが簡単に作成できるグラフィックパターンエディタ●MIDI対応のX-BASIC。

※1 アプリケーションソフトおよび周辺機器のうち、一部動作しないものがあります。詳しくはシャープお客様相談窓口にお問い合わせください。

※2 10.4型カラー液晶ディスプレイLC-10C1-H標準価格598,000円(税別)、接続ケーブルAN-1515X標準価格4,200円(税別)をご使用ください(SX-WINDOW対応アプリケーションのみ。色数に制限があります)。

※3 別売のMIDIインターフェイスが必要です。

5.25"FDDマンハッタンシェイプシリーズ

■X68000伝統のマンハッタンシェイプを継承 ■5.25インチFDD2基搭載 ■80MBハードディスク内蔵(CZ-510C)※ ■マウス・トラックボール標準装備 ■ASCII準拠フルキーボード採用

※CZ-500Cには、2.5インチ80MB内蔵用ハードディスクドライブCZ-5H08(近日発売)/2.5インチ160MB内蔵用ハードディスクドライブCZ-5H16(近日発売)を用意しています。

X68030
32bit PERSONAL WORKSTATION

本体+キーボード+マウス・トラックボール
5.25インチFDDタイプ CZ-500C-B(チタンブラック)近日発売
HDDタイプ CZ-510C-B(チタンブラック)近日発売
14型カラーディスプレイ
CZ-608D-B(チタンブラック)標準価格94,800円(税別・チルトスタンド同梱)

3.5"FDDコンパクトシリーズ

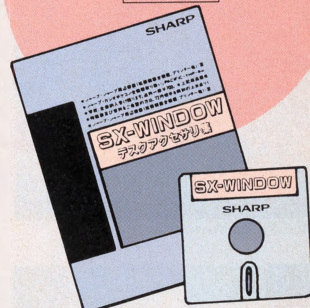
■32ビットのハイパワーを凝縮したコンパクトフォルム ■3.5インチFDD2基搭載 ■80MBハードディスク内蔵(CZ-310C)※ ■マウス標準装備 ■コンパクトキーボード採用

※CZ-300Cには、2.5インチ80MB内蔵用ハードディスクドライブCZ-5H08(近日発売)/2.5インチ160MB内蔵用ハードディスクドライブCZ-5H16(近日発売)を用意しています。

Compact

本体+キーボード+マウス
3.5インチFDDタイプ CZ-300C-B(チタンブラック)近日発売
HDDタイプ CZ-310C-B(チタンブラック)近日発売
14型カラーディスプレイ
CZ-608D-B(チタンブラック)標準価格94,800円(税別・チルトスタンド同梱)

NEW
X68030
32bit PERSONAL WORKSTATION
デビュー

春一番
プレゼントセール
実施中!

X68030シリーズを
お買い上げいただき、
EXEクラブにご入会の方
先着1,000名様に

〔EXEクラブ入会申し込みハガキは本体同梱〕
既入会の方も、X68030の新会員NO.で新規登録します。

SX-WINDOWデスクアクセサリ集 内容

■SX-WINDOWの環境をより良くするツール
- ウィンドウアイコニファイ
- ファイルサーチ
- キーノート
- スクラップブック
- ソフトウェアキーボード
- フォントリンカ

■楽しめるツール
- スクリーンセーバ
- ミュージックボックス
- パズル

■実用的なツール
- 電子手帳通信ツール(DB-Z、PV-F1に対応)
- アドレス
- スケジューラ

■お問い合わせは…電子機器事業本部システム機器営業部 〒545 大阪市阿倍野区長池町22番22号 ☎(06) 621-1221(大代表)
電子機器事業本部AVCシステム事業推進室 〒162 東京都新宿区市谷八幡町8番地 ☎(03)3260-1161(大代表)
シャープ株式会社

資料請求券 X68030 Oh!X 3月

新登場 32ビットマシンX 68030

MIRAGE System Model Stuff

究極タイガー

チェルノブ

THE USER'S WORKS

DōGA CGアニメーション講座

Oh!X

CONT

●特集

〈スタッフ〉
●編集長/前田 徹 ●副編集長/植木章夫 ●編集/浅井研二 山田純二 豊浦史子 ●協力/有田隆也 中森 章 林 一樹 吉田幸一 華門真人 吉田賢司 影山裕昭 大和 哲 村田敏幸 丹 明彦 三沢和彦 長沢淳博 宮島 靖 金子俊一 浦川博之 石上達也 柴田 淳 御木徳高 瀧 康史 ●カメラ/杉山和美 ●イラスト/山田晴久 寺尾響子 高橋哲史 川原由唯 ●アートディレクター/島村勝頼 ●レイアウト/元木昌子 ADGREEN ●校正/グループごじら

表紙絵：塚田　哲也

E　N　T　S

1993 MAR.
3

■広告目次

成熟するウィンドウ環境で

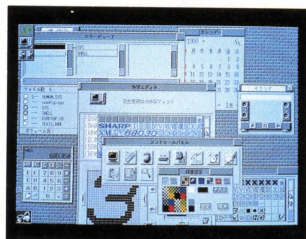
65,536色対応、動画ウィンドウ標準装備。

SX-WINDOW ver3.0

NEW

CZ-294SSD　3月発売予定

512×512ドットのエリア内で、自然描画に迫る美しい表現が可能な65,536色表示のグラフィックウィンドウを駆使できます。さらにグラフィックウィンドウ内でのアニメーション動画表示、各種グラフィックデータのコンバートも実現しました。高機能エディタ「日本語マルチフォントエディタ」を標準装備。アウトラインフォントの展開もフォントマネージャの効率化により、さらに高速化が図られています。その他、最大ズームサイズの設定や任意サイズのグラフィックを背景に設定できるなど、クリエイティブワークをサポートする数々の便利機能を装備しています。Human 68k ver3.0 および ASK68K ver3.0を標準装備しています。

※メインメモリ2MB以上必要です。※SX-WINDOW ver1.0/1.1/2.0をお持ちの方には有償バージョンアップを行います。

(日本語マルチフォントエディタの特長)

■**自由なフォント設定:**フォントタイプ、サイズ、スタイルを文字単位に指定可能。ルビも自由な大きさで付けられます。■**ワープロ機能:**禁則処理(追い出し、ぶら下がりも指定可能)、ワードラップ(半角文字)。■**ユーザーカスタマイズ機能:**キー割り当て、マクロ定義、メニュー定義(アイコンも定義可能)、外部コマンドなど。■**イメージデータの貼り付け:**パターンエディタなどで作成したビットイメージデータの貼り付けが可能。■**シングルウィンドウモードの追加:**複数のファイルをひとつのウィンドウで編集ができます。ファイルごとに編集環境の切り換えが可能。■**その他:**レイアウト機能の強化、矩形カット&コピー/矩形ペースト、マーク・ジャンプ機能。

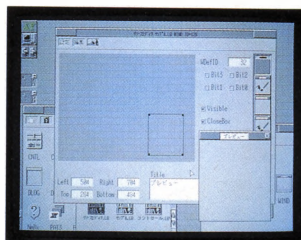
待望のSX-WINDOW開発支援ツール。

SX-WINDOW 開発キット Workroom SX-68K

NEW

CZ-288LWD　3月発売予定

SX-WINDOW用のソフト開発に必要な開発ツールやサンプルプログラムを装備。プログラムの編集、リソースの作成、コンパイル、デバッグといった一連の作業をSX-WINDOW上で効率よく実行できます。初めてSX-WINDOW用のプログラムに挑戦する人にも、簡単に基本機能の理解ができる33種のサンプルプログラム付き。また各マネージャ解説と関数リファレンスの詳細なマニュアルも装備しています。

※本ソフトのご使用に際しては、メインメモリ4MB以上、SX-WINDOW ver2.0以上、C compiler PRO-68K ver2.1が必要です。

キット構成

■**開発ツール**

●**SXデバッガ**
SX-WINDOW上で複数のプログラムを同時にデバッグすることができるソースコードデバッガ。

●**リソースエディタ**
SX-WINDOW上のリソースをリソースタイプごとの編集ウィンドウでビジュアルに作成・編集が可能。

●**リソースリンカ**
Cコンパイラやアセンブラで作成したリソースデータファイル(オブジェクトファイル)をリンクしてリソースファイルを作成。

●**サンプルメイク**
サンプルプログラムのコンパイル作業をSX-WINDOW上から、XCver2.1のMAKE.Xを呼び出して、自動実行する簡易メイクユーティリティ。

■**サンプルプログラム**

●**基礎編(23種)**
各マネージャの基本的な機能のみを用いた基本動作の理解。

●**応用編(4種)**
基礎編での基本機能を応用した簡単なアプリケーションの作成。

●**実用編(6種)**
基礎/応用編での機能を駆使した、実用的なアプリケーションの作成。

■**その他ファイル**

●**インクルードファイル**
Cコンパイラとアセンブラ用の関数定義、データ定義ファイル。

●**ライブラリファイル**
Cコンパイラ用関数ライブラリ。

マニュアル
●ユーザーズマニュアル●プログラマーズマニュアル●SXライブラリリファレンスマニュアル

資料請求券 CZソフト Oh!X 3体

さらに高度な創造次元へ。

●多彩なサウンドクリエイトを実現するFM音源サウンドエディタ。

SOUND SX-68K

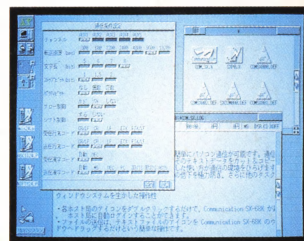
CZ-275MWD　標準価格15,800円(税別)

他のミュージックソフトで演奏中の音色を、簡単に作成、変更ができるマルチタスク機能、またエディット、イメージ、ウェーブの3つの編集/確認モードを装備。作成中の音色も50曲の自動演奏でリアルタイムに確認、編集できます。まさにミキサー感覚で音創りが楽しめるツールです。 (2MB、ver1.1)

●マルチタスク機能をはじめ、通信環境がさらに充実。

Communication SX-68K

CZ-272CWD　標準価格19,800円(税別)

通信環境をさらに高めたウィンドウ対応の通信ソフトです。マルチタスク機能により他のアプリケーションソフトを実行中でも簡単に通信が可能。また、ホスト局をクリックするだけの自動ログイン機能、初心者にも簡単なプログラム機能、最新モデム(20種類)もフルサポートしています。 (2MB、ver1.1)

●ウィンドウ対応グラフィックツール。

Easypaint SX-68K

CZ-263GWD　標準12,800円(税別)

マウスによる簡単操作、65,536色中16色の多彩な表現、クリエイティブマインドに応えるウィンドウ対応ペイントツールです。同時に複数のウィンドウを開いて編集でき、各ウィンドウ間でのデータ交換もできます。

(2MB、ver1.1)

●「SX-WINDOW開発キット」のサポートツール。

開発キット用ツール集

CZ-289TWD　開発中

SX-WINDOW開発キットをさらに使いやすくするためのツールです。SXコールの簡易リファレンスを簡単に検索する「インサイドSX」、イベントの発生を常時監視確認するイベントハンドラ、リアルタイムにメモリブロックの利用状況を表示するヒープビューアなど11種のツールが用意されています。

(2MB、ver2.0)

※(2MB、ver1.1)の表示は、メインメモリ2MB以上、SX-WINDOW ver1.1以上が必要であることを示します。

充実の PRO-68K シリーズ

●マルチフォント印字に対応。

Multiword ver2.0 NEW

CZ-225BSV

標準価格32,000円(税別)

Zeit社の書体倶楽部をサポート。同時に6書体のフォントが指定可能、レーザプリンタのフォントも複数使用できます。またキー操作やメニューの改良、均等割り付け、グラフィックのアイコン化なども可能。

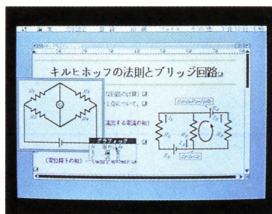
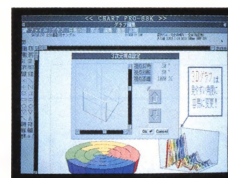
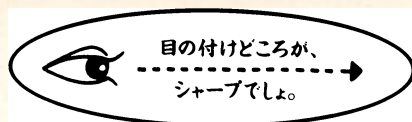
※MultiwordおよびMultiword ver1.1をお持ちの方には有償バージョンアップを行います。

●ビジネスグラフチャート。

CHART PRO-68K

CZ-267BSD　標準価格38,000円(税別)

※以上のPROシリーズのソフトの動作にはメインメモリ2MB必要です。

※発売予定のソフトの画面写真は実物とは異なる場合があります。

●お問い合わせは…シャープ㈱電子機器事業本部AVCシステム事業推進室　〒162 東京都新宿区市谷八幡町8番地 ☎(03)3260-1161(大代表)へ　シャープ株式会社

SHARP

“感性”咲かせるワ

POWER WORKSTATION

インテリジェントなパフォーマンスを誇るX68000Compact XVIと多彩にラインアップされたペリフェラル。感性を刺激するクリエイティブなワークステーション環境が自在に構築できます。

- パーソナルワークステーション(2HD3.5インチFDDタイプ・本体＋キーボード＋マウス)
  CZ-674C-H(グレー) 標準価格**298,000**円(税別)
- 15型カラーディスプレイテレビ
  CZ-614D-TN(チタンブラック)・**-BK**(ブラック) 標準価格**135,000**円(税別)
  ■ディスプレイテレビ/CZ-6TU用RGBケーブル**CZ-6CR1** 標準価格**4,500**円(税別)
  ■ディスプレイテレビ/CZ-6TU用TVコントロールケーブル**CZ-6CT1** 標準価格**5,500**円(税別)
- 80MB内蔵用ハードディスクドライブ
  CZ-68HA 好評発売中
- 5.25インチ増設用フロッピーディスクドライブ
  CZ-6FD5 標準価格**99,800**円(税別・接続ケーブル同梱)
- 光磁気ディスクユニット
  CZ-6MO1 標準価格**450,000**円(税別)
  ■SCSI変換ケーブル**CZ-6CS1** 標準価格**12,000**円(税別)
- 2MB増設RAMボード
  CZ-6BE2D 標準価格**54,800**円(税別・取り付け費別)
  ■2MB増設RAM**CZ-6BE2B** 標準価格**54,800**円(税別・取り付け費別)×2
  ■数値演算プロセッサ**CZ-6BP2** 標準価格**45,800**円(税別・取り付け費別)
- 48ドット熱転写カラー漢字プリンタ
  CZ-8PC5-BK(ブラック) 標準価格**96,800**円(税別)
- MIDIボード
  CZ-6BM1A 標準価格**26,800**円(税別)
- インテリジェントコントローラ
  CZ-8NJ2 標準価格**23,800**円(税別)

(68買ったら EXEクラブへ 入ろう!)

EXEクラブって何だ?

X68000を手に入れたら、やっぱり他のユーザーがどんな風に使っているのか気になるもの。ということでEXEクラブは、そんなあなたのための、他の68ユーザーとのコミュニケーションをバックアップする、情報交換の場です。

資料請求券
X68000ペリフェラル
Oh!X
3係

GRAPHIC WORKSTATION

- パーソナルワークステーション(2HD3.5インチFDDタイプ・本体＋キーボード＋マウス)
  CZ-674C-H(グレー) 標準価格**298,000**円(税別)
- 21型カラーディスプレイ **CU-21HD** 標準価格**148,000**円(税別)
- 80MB内蔵用ハードディスクドライブ **CZ-68HA** 好評発売中
- 光磁気ディスクユニット **CZ-6MO1** 標準価格**450,000**円(税別)
  ■SCSI変換ケーブル**CZ-6CS1** 標準価格**12,000**円(税別)
- 2MB増設RAMボード **CZ-6BE2D** 標準価格**54,800**円(税別・取り付け費別)
  ■2MB増設RAM**CZ-6BE2B** 標準価格 **54,800**円(税別・取り付け費別)×2
  ■数値演算プロセッサ**CZ-6BP2** 標準価格**45,800**円(税別・取り付け費別)
- カラーイメージスキャナ
  CZ-8NS1 標準価格**188,000**円(税別)
  ■スキャナ用パラレルボード**CZ-6BN1** 標準価格**29,800**円(税別)
- カラーイメージジェット
  IO-735X-B(ブラック) 標準価格**248,000**円(税別)
  ■接続ケーブル**IO-73CX** 標準価格**5,500**円(税別)

STANDARD WORKSTATION

- パーソナルワークステーション
  (2HD3.5インチFDDタイプ・本体＋キーボード＋マウス)**CZ-674C-H**(グレー) 標準価格**298,000**円(税別)
- 14型カラーディスプレイ**CZ-608D-H**(グレー) 標準価格**94,800**円(税別)
- 5.25インチ増設用フロッピーディスクドライブ**CZ-6FD5** 標準価格**99,800**円(税別・接続ケーブル同梱)

TFT COLOR LCD WORKSTATION

- パーソナルワークステーション
  (2HD3.5インチFDDタイプ・本体＋キーボード＋マウス)**CZ-674C-H**(グレー) 標準価格**298,000**円(税別)
- 10.4型カラー液晶ディスプレイ**LC-10C1-H**(グレー) 標準価格**598,000**円(税別)
  ■接続ケーブル**AN-1515X** 標準価格**4,200**円(税別)

※カラー液晶ディスプレイを接続してご使用の場合、SX-WINDOW上のアプリケーション利用に限定されます。

本体同梱の入会申込ハガキを送るだけで、自動的に無料入会。さらに下記の特典付き。

メリット1 会員ナンバー入りオリジナル会員電卓がもらえる。

メリット2 各種フェアご優待・イベント案内等、数々の特典がある。

●お問い合わせは…
シャープ株式会社
電子機器事業本部システム機器営業部
〒545 大阪市阿倍野区長池町22番22号☎(06)621-1221(大代表)
電子機器事業本部AVCシステム事業推進室
〒162 東京都新宿区市谷八幡町8番地☎(03)3260-1161(大代表)

brother

ILLUSION CITY 幻影都市

©1991 MICRO CABIN CORP.

科学が創る影なる都。

ILLUSION by YUKIOKITTA/CHARACTER DESIGN by 百鬼丸

禍々しき気に満ちた近未来都市、香港。狂気と悪しき欲望とが渦巻くこの都市(まち)を、
ける。失われた己の過去を求めて、迫り来る危険に自ら身を投じる男、対魔掃討(ダイバー)
の対魔特別攻撃班(デモンバスター)に属する女、"美紅"と共に、その実体さえ知れぬ巨大な悪
愛用の銃(ブラスター)を放つ。果てしなく続く戦いの日々は、いつしか眠ることさえ忘れさせて

人民警察(アウターポリス)
軍身の気を込めて
……。

3月28日発売予定

サイバーパンク!! RPG「幻影都市」

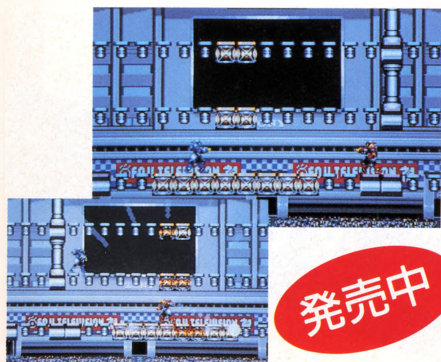
■8等身キャラクタ採用 ■キャラクタ演出革命!!
■ジョイパット&マウスオペレーション可能
■VRシステムVer.2.5搭載 ■MIDI対応

TAKERU価格 ¥6,800(税込)
■対応機種/X68000版
■制 作/TAKERUソフト

©マイクロキャビン

ストライクレンジ

サイドビュー、縦横スクロールのロボット対戦シューティングアクション、何層もの床で構成された近未来スタジアムで、今、最も危険なスポーツが始まった!ロボットの種類は8体、2人対戦モード付き、迫熱興奮のバトルに挑戦だ!

発売中

TAKERU価格 ¥4,800(税込)
■対応機種/X68000版
■制 作/ギミックハウス

機甲装神 ヴァルカイザー

近未来、エネルギーを増幅する人工外皮「バイオローダー」の研究に伸びる黒い魔手。岬博士と妹留奈に襲いかかる者達の正体は…?
美少女とメカとアニメーションといえば、ご存知「サイレンス」!初めてのX68000移植版がついに登場!もちろんフルアニメーションが、ガンガン入ってます!!

発売中

TAKERU価格 ¥4,800(税込)
■対応機種/X68000版
■制 作/サイレンス

ダイナマイト・デューク

あの名作がTAKERUで再び!!

- 大人気アーケードゲームから移植!
- シューティングの興奮+アクションの一体感!
- アメリカン・コミック調の美しい画面
- アニメのように派手に動く、ド迫力デカ・キャラ!
- 変化に富んだ各ミッションと個性あふれる敵キャラ
- 必殺の一撃「ダイナマイト・パンチ」の快感!

発売中

TAKERU価格 ¥3,800(税込)
■対応機種/X68000版
■制 作/ヘルツ

ブラザー工業株式会社 〒467 名古屋市瑞穂区苗代町2番1号
TAKERU事務局 (052)824-2493
東京営業所(03)5203-7133
大阪営業所(06)252-4234

通信販売
通信販売をご希望の方は、ソフト名・機種名・住所・氏名・電話番号を明記の上TAKERU事務局まで現金書留でお申し込み下さい。
代金引換は一度、現金書留で申し込んで頂いた方に案内させて頂きます

The スーパーファミコン

3/5号

全国の書店、コンビニエンスストアにて発売!

ドラゴンクエストI・II

特集開幕!スポーツ大特集

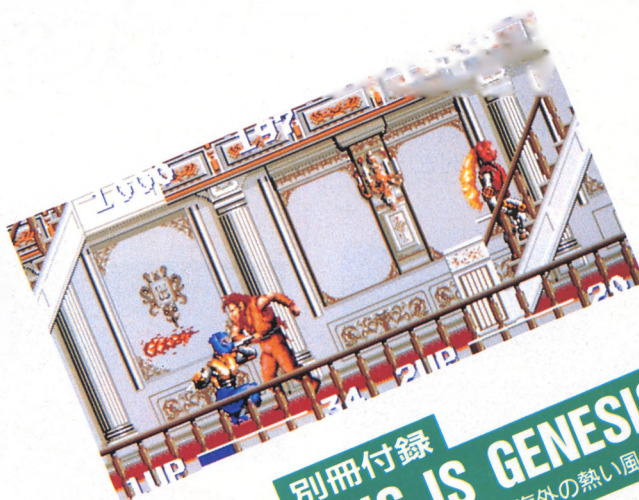
～今シーズンの行方とおススメSFCスポーツゲームを紹介～

すぎやまこういちのゲーム源流記(後編)ゲスト・植松伸夫

別冊付録「スターフォックス」読本

予告! Theスーパーファミコン臨時増刊号 3月25日発売予定 シューティング大特集号

特集 産地直送!

特報!

別冊付録 THIS IS GENESIS

寒い冬を吹き飛ばす海外の熱い風

SOFT BANK

お近くの書店でお買い求めください

ついに登場！

32ビットになったX68000の最上位機種

X68030

マンハッタンシェイプのツインタワーが32ビットになってそびえ立つ。くぅ～っ，赤バッジが眩しいぜ！
CPUにモトローラの良心，MC68EC030を搭載。動作クロックは25MHz。チキショ～，速いじゃないか！
おっ，こっちの小さいのはなんだ，Compactじゃないのか。エエッ!?　Compactも32ビットか。
X68000の超多機能をフル装備で，ハードディスクも内蔵，メモリも本体内に12Mバイトまで入るなんて。シャープめ～。

CZ-500C/510C
伝統のマンハッタンシェイプに赤いロゴが映える5.25″FDDモデル。32ビットパワーで新たなステータスシンボルに！

初代X68000が発表されてから丸6年。ユーザーの期待を一身に背負い，ついに32ビット機が登場した。思わず気が動転してしまうほどうれしいビッグニュースだ。
発表されたのは，68EC030/25MHzを搭載した，その名もX68030。お馴染みツインタワーの5.25″FDDタイプCZ-500C/510Cが3月10日，ちょっと遅れてコンパクトな3.5″FDDタイプCZ-300C/310Cが5月14日に発売となる。510C/310Cは80Mバイトのハードディスク内蔵モデルだ。
スプライト，グラフィック，サウンド，そして数えだしたらきりがないほどの贅沢なハードウェアをそっくり継承しつつ，強力なCPUでドライブする。さらに強化されたHuman68k Ver.3.0，SX-WINDOW Ver.3.0で使いやすさも一段と向上した。

●68030/25MHzの威力

なんといってもCPUから。68EC030というのはもちろんモトローラの32ビットCPU。いやMPUとお呼びすべきか。通常の68030と違うのはMMU（メモリマネージメントユニット）がないことだけ。OSの仕様やメモリの使用状況からみて，妥当な選択だろう。もちろん処理能力は68030となんら変わらないスグレモノである。
動作クロックは25MHzで，メモリアクセスは清く正しく32ビットバスによる。ウエイトは1つくらい入るでしょうとのこと。まあクロックが25MHzというと，通常のメ

CZ-300C/310C
機能美を追求した3.5″FDDモデルのコンパクトタイプもチタンブラックで新登場。機能はCZ-500C/510Cとまったく同じだ！

モリが追いつく速度ではない。もちろん回路設計では，なるべくウエイトが入らないような配慮がされるわけで，通常のメーカーならまずノーウエイトといってしまうところだ。生まじめなシャープではノーウエイトとはいいたくないらしい。ただ，68030には内部キャッシュがプログラム用とデータ用にそれぞれ256バイトずつあり，ある程度の効果が期待できる。

さあ，どれくらい速いか気になるところだろう。シャープ発表の公称値でいうと，Dhrystone比（データキャッシュ，プログラムキャッシュ共にON）で，

X68000(10MHz)の4.3倍

X68000XVI(16MHz)の2.4倍

ということだ。これはCPUの演算速度の比較だから，実際の使用においては用途によって値は変わってくる。また，グラフィックが動くようなプログラムを目のあたりにすると体感速度は数字以上。脅威のパワーを実感できるはずだ。

●コプロセッサは68882に対応

コプロセッサとしてはオプションで数値演算プロセッサCZ-5MP1が用意されており，メイン基板に専用ソケットがある。これはモトローラの68882で，チップだけを買ってきてソケットに装着すればよい。従来のX68000の数値演算プロセッサは68881で，これをFLOAT3.Xなどのデバイスドライバ経由で利用していた。しかし，X68030の場合，68030のコプロ命令を使って直接68882をドライブすれば，従来とは桁違いの速度で実数演算を行うことが可能である。

● 4 Mバイト標準装備

メインメモリは4Mバイトを標準装備。本体内に最大12バイトまで増設可能となっている。ただし，それ以上の増設はいまのところできない。増設メモリは例によって2階建て。まず4Mバイトの増設RAMボードをさし，さらに増設したい場合は4MバイトのRAMモジュールを重ねるという仕組みだ。SIMMではないが，この件に関するシャープの見解は「品質にバラつきのあるSIMMでは動作保証ができないためX68000には使いたくない」というものだ。Macintoshや IBM互換機では安いSIMMをボコボコ使っているが，X68000はあくまでクオリティ重視のパソコンなのだ。なお，増設RAMはX68030の4機種に共通だ。

●2.5″ハードディスク

また，CZ-510C/310Cには2.5インチの80Mバイトハードディスクが標準装備されている。CZ-500C/300Cでもあとから内蔵可能だ。オプションの内蔵用ハードディスクは80Mバイトと160Mバイトのものが予定されている。

●3.5″FDDは 3 モードに

X68000 Compactの3.5″FDDは1.2Mの2HDしか読めなかったが，X68030ではオートイジェクト回路を変更し，720K，1.44Mフォーマットにも対応した。また，内蔵ROMのIOCSコールも2HD以外のデータディスクが読めるよう拡張されている。

●拡張スロット

背面の拡張スロットは2つあり，従来互換の16ビットバス/10MHzで，X68000のボード類が利用できる。メモリなどを外部にさすと遅くなってしまうが，本体内に最大値の12Mバイトまでフル実装できるので問題ないだろう。MIDIボードのようなX68000ユーザーにとっての必需品がそのまま使えるのはやはりありがたい。

●気になるソフトの互換性

ソフトの互換性はどうだろう。当初の予想ではX68000用のかなりのソフトが動かなくなるのではと心配されていたが，予想に反して動くものが多いようだ。シャープでは6～7割のソフトがX68030でも動作するとしている。また，動かないものでも多少の手直しで動作するようになるのではないかと思われる。

一般に68030は68000の上位互換であるといわれているが，実際に互換性が保証されているのはユーザーモードで動作するプログラムの場合で，スーパーバイザモードで動くものは保証されていない。Macintoshでソフトの互換性があるのは，Macintoshには特別なハードウェア機能がほとんどなく，ソフトはいずれもMac OSの上で動くものだからだ。

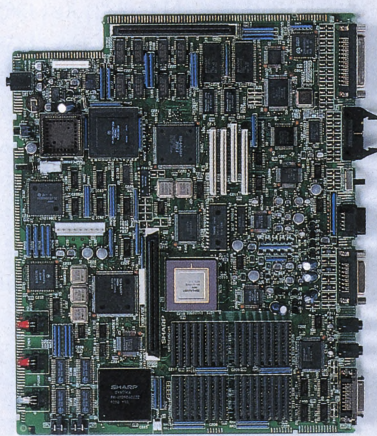
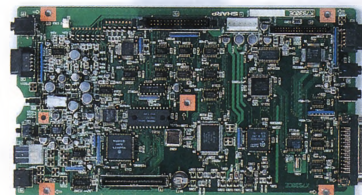
また，CPUが上位互換でも，周辺のハードウェアが変更されたら，ハードを直接アクセスすることの多いX68000のソフトはまず動かなくなるところだ。幸いハードの変更はほとんどなく，かなりのソフトがそのまま利用できる。たとえば，コナミの名作“パロディウスだ！”やエグザクトの“ナイアス”のようなX68000のハードウェアをとことん使い込んだプログラムが32ビットのX68030でも動く。これはすごいことだ。Oh!Xではソフトの動作チェックも行う予定である。

さて，使用するソフトによってはX68030の25MHzでは速すぎて都合が悪いこともあるだろう。X68030ではクロックの切り替えスイッチはないが，XF1，XF2キーを押しながら起動すると，それぞれX68000(10MHz)，X68000(16MHz)相当のスピードで動作するようになる。

●Human68k Ver.3.0

また，従来のHuman68k(Ver.2.03以前のもの)は使えないが，ディスク起動のゲームなどを利用するために互換OS(Human68k Ver.2.15)がROMで用意されている。詳し

表1　X68030 製品一覧

型番	FDD	HDD	標準価格	発売日
CZ-500C	5.25″	−	398,000円	3月10日
CZ-510C	5.25″	80MB	488,000円	3月10日
CZ-300C	3.5″	−	388,000円	5月14日
CZ-310C	3.5″	80MB	478,000円	5月14日

CZ-300C/310Cのメインボード

CZ-500C/510Cのメインボード

65536色グラフィック表示やアニメーションのための動画管理マネージャなど，一段と強化されたSX-WINDOW Ver.3.0。付属のエディタも機能が一新された

くは，あとの記事をご覧いただきたい。

X68030のシステムは32ビット対応になるなど，大幅にバージョンアップされ，Human68k Ver.3.0になった。

また，日本語処理機能のASK68KもVer.3.0になった。実に4年ぶりのバージョンアップだ。変換アルゴリズムや，品詞情報が大幅に見直され，変換効率が格段にアップ。処理速度も向上している。そして，辞書メンテナンスがSX-WINDOW対応になったのもうれしい。

●SX-WINDOW Ver.3.0

そして忘れてはならないのが，さらに強力になったSX-WINDOW Ver.3.0だ。デスクトップ上の任意の位置にグラフィックウィンドウ(最大512×512ドット)を開くことができ，その中では65536色表示が可能だ。また，TIFFやJPEGなど，画像データの圧縮/伸長をシステムでサポート。動画と音楽を同期させる時間管理もマネージャとして用意している。

SX-WINDOW上のアクセサリだったエディタ.Xもワープロ並みに強化されている。任意の範囲に対して，自由に書体や行間隔を設定でき，パターンエディタで作ったグラフィックも貼り込める。下手なワープロよりもずっと強力な仕様となった。

SX-WINDOWは世界に誇れる素晴らしいウィンドウシステムといえるだろう。プラットフォームとしてのSX-WINDOWを発展させていけば，将来X68000の次世代マシンが登場してもSX-WINDOWの環境を継承できるはずだ。シャープには今後もSX-WINDOWのアプリケーションや周辺デバイスドライバ類を充実させていってもらいたい。

やっぱり32ビットマシンだと高価なカラー液晶ディスプレイもいっそう似合うぞ。う～ん，贅沢をもつ喜び

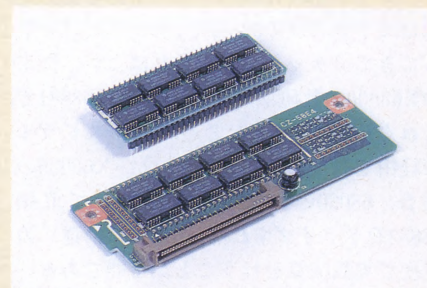
●X68030のパフォーマンスに期待

ちょっと高いのではといわれるX68000だが，今回のX68030はコスト的にもかなり無理をした価格設定になっているようだ。

ちょっと比較できるマシンがないが，CPUとクロックが同じものを強いて挙げるなら，Macintosh IIciぐらいか。かなり定価が引き下げられたとはいえ，1月現在，FDDモデルが698,000円とかなり高い。

逆に同一価格帯ではPC-9821Ae(486SX/25MHz)あたり。3.5″FDDモデルが358,000円で3万円安といったところだが，装備を比較すればX68030のほうが十分にコストパフォーマンスは高いように思われる。もちろんアプリケーションの数を比較されると勝負にはならないが，Windowsでも使おうというのならAeでも十分とはいえない。せめてAs(486DX/33MHz，448,000円)以上の機種が必要になる。

逆にSX-WINDOWならX68030で申し分のない操作環境が得られるだろう。

少なくともX68000を使っているユーザーならX68000が高いパフォーマンスをもつことを実感しているはずだ。そして値段では語れない魅力があることも忘れてほしくない。あらゆるパソコンのなかでも，多機能，高品質，そして使いやすさを誇るパーソナルマシン，それがX68000だ。そのX68000が32ビットになり，どれだけのパフォーマンスを発揮するのか，製品の発売を楽しみに待ちたい。

増設RAMボード(CZ-5BE4)とモジュール(CZ-5ME4)

数値演算プロセッサ(CZ-5MP1)

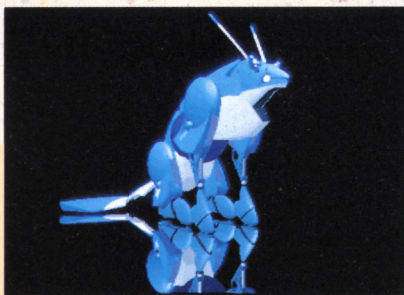
表2

周辺機器(内蔵用)	型番	標準価格	発売日
4MB増設RAMボード	CZ-5BE4	54,800円	3月25日
4MB増設RAMモジュール	CZ-5ME4	49,800円	3月25日
数値演算プロセッサ	CZ-5MP1	54,800円	3月25日
80MBハードディスク	CZ-5H08	未定	近日発売
160MBハードディスク	CZ-5H16	未定	近日発売

2月号で作ったコースとF1のみのカット

BGMAKEの背景画と2月号の画像を合成

看板や芝生，ガードレール，そのほかの部品をすべて置き，仕上げをして完成させた画像

CGAコンテストはただいま選考中ですが，連載記事中で触れている作品のなかから，主なものを1コマずつ紹介しておきましょう。
（敬称略）

「OBJECT:MECHANICAL HOUND」下岡正道

「ハッピー バレンタイン」客野優子

「MISSION」浅野英史

「ふしぎなえんとつ」藤村典由

年賀状だよ！
Oh!X reader'sぎゃらりぃ

あけましておめでとうの巻

毎年恒例の年賀状紹介コーナーです。ちょっと時期外れな気もしますが，そんなことを気にしちゃいけません。カラフルなイラストを楽しみましょう。

▲小川　裕美（山口県）

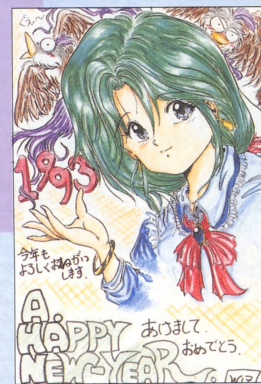
▲岡村　直也（兵庫県）

▲節政　暁生（千葉県）

▲占部　哲彦（広島県）

▲武田　和凡（京都府）

▲佐藤　一秀（愛知県）

▶玉野　健一（奈良県）

▲横井　賢一（富山県）

▲大岩　道明（千葉県）

スタッフからの年賀状

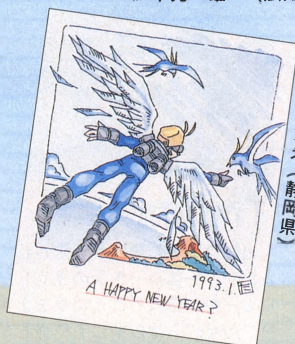
▲YAMADA JUNJI

◀KAWAHARA YOUI

▲狩郷　秀毅・由美（愛媛県）

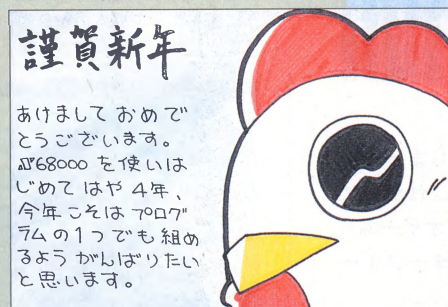
▲中光　雄二（広島県）

▲鈴川　美佳子（東京都）

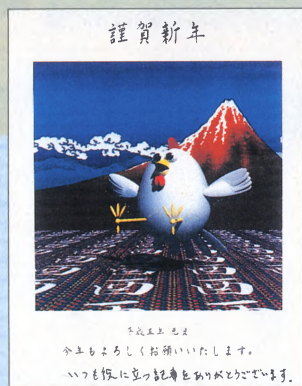
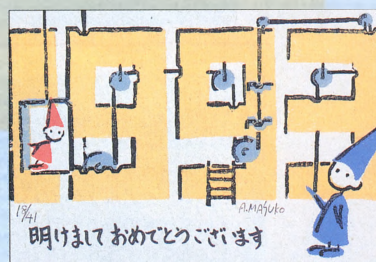
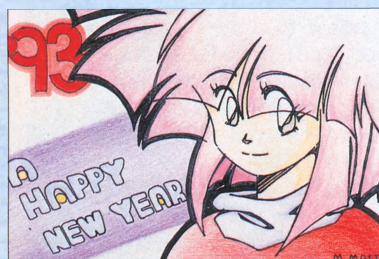
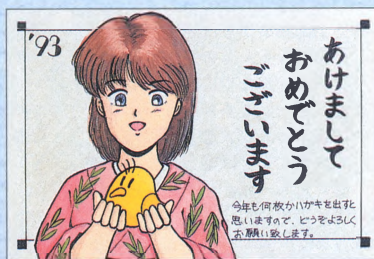
▲加藤　隆（佐賀県）

▲鈴木　貴久（静岡県）

▲阿部　雅敏（大阪府）

▲石田　伯仁（神奈川県）

▲岡田　徹（長崎県）

▲阿部　哲也（兵庫県）

▲吉岡　洋明（埼玉県）

▲曽谷　修二（兵庫県）

▲石田　博也（茨城県）

▲益子　暁（東京都）

▲前田　基行（兵庫県）

▲尾形　雅治（広島県）

▲薮田　俊平（和歌山県）

お知らせ

スタッフを除く25名の皆様には，記念品を差し上げます。そして，次の「Oh!X reader'sぎゃらりぃ」は，5月号の「言わせてくれなくちゃだワ」で行う予定です。それまでに，腕をびしばし磨いておきましょうね。

SOFTWARE INFORMATION

なんか寂しい新作情報となってしまったが，新しいマシンも出たことだし，これからに期待しよう。あ，ちなみに“ビデオゲーム・アンソロジー”の第3弾は「スターフォース」らしいぞ。

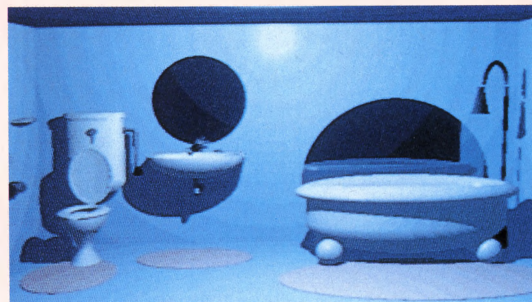
Traum

当初は春頃に発売が予定されていたこのゲームだが，一部ルーチンの見直しなどが入ったらしく，少し遅れることになるようだ。

このゲームは以前にも紹介したように，泡をつなげて，ネズミがゴールまでたどりつけるように道を作ってやるのが目的。こう聞くと簡単なようだが，実際に作るとなるとさまざまな要素が絡み合い，開発も容易にはいかないだろうことは想像できる。さらに奥行きもつくとなったら，なおさらである。

ネズミはちゃんと3匹個別の思考ルーチンをもつような感じだし，敵キャラも個性的な動きを見せてくれそうだ。面ごとにユニークな仕掛けも期待できる。まあ，じっくりと作って，じっくりとバランスをとり，楽しいアクションゲームに仕上げてほしい。

掲載した写真のうち2枚は，未完成面の背景部分のみである。すべての面は一軒の家を構成する部屋ということになるので，シャワールームなどもあるようだが，シャワーや，トイレの水が流れたりするのだろうか。

X68000用　5″2HD版　価格未定
M.N.M Software　☎0423(60)3084

オーバーテイクひとり旅状態

1.	オーバーテイク	1
2.	ファイナルファイト	4↑
3.	ふしぎの海のナディア	3
4.	ポピュラスII	5↑
5.	エトワールプリンセス	7↑
6.	三國志III	−↑
7.	グラディウスII	9↑
8.	ストライダー飛竜	8
9.	ロードス島戦記II	−↑
10.	ジェノサイド2	−↑

今月も「オーバーテイク」はぶっちぎり。2位から10位の票を全部集めても「オーバーテイク」にかなわないんだから，その強さは並じゃない。ハガキを見ると，デモや広告による雰囲気づくりのうまさ，ゲームを盛り上げるサウンドエフェクトを評価する人が大部分でした。期待していたゲームスタイルとは違ったという声もありますが，必ずしもそれがマイナス評価にはなっていないようですね。

それを追いかけるゲームがちょっと見当たらない，そのほかのランキング。期待の「ストライダー飛竜」は意外にも8位どまり。マニアックな内容は，ゲームセンターよりも家庭で評価されると思われたのですが。

その他の作品は前回のランクをほぼ維持していますが，目立つのが「三國志III」の伸び。いままで何度もランクイン/アウトを繰り返しているので，光栄のファンは気まぐれなんですね，きっと。

このTOP10では，1位が独走した月には得票が荒れるという習性がなぜかあります。今月も上位陣が平穏そうに見えるなか，実は先月2位の「ムーン/テラクレスタ」がランクアウトしています。この寿命の短さはいったい……。電波ファンは薄情なのでしょうか？

それから，下のほうでは「ロードス島戦記II」や「ジェノサイド2」が返り咲き。固定ファンが頑張ってハガキを書き続けた甲斐がありましたね。地味ながら長く評価されるゲームが入ってくるのもうれしい話。じゃ。　(浦)

来月からこのTOP10のコーナーは，新作ソフトの前評判をランキングします。アンケートハガキの“期待している新作ソフト”の欄を集計するというわけです。皆さん，よろしくね。

メガロマニア

このゲームは1992年3月号の“TREND ANALYSIS”で取り上げている。もともとはAMIGAのゲームである。ジャンルはリアルタイム戦略シミュレーション，自分の部族を操って相手の部族をやっつける。「なんか“ポピュラス”みたい」と思うだろうが，このゲームが「ポピュラス」に強く影響を受けているのは確かだ。

プレイヤーができるのは，特定の武器を作らせること，自分の部族の兵隊の配置（攻撃および防御），開発/採掘などへの人間の振り分けなど。そして，このゲームでいちばん特徴的なのは，作ることのできる武器が技術レベルに制限されている点である。つまり，技術レベルがBC9500（原始時代）なら石と棒しか使えず，AD2001（宇宙時代）ならUFOやSDI（なんか懐かしい響き）まで使える。このギャップにはなんともいえないおかしさがある。

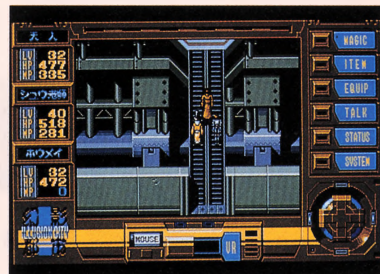
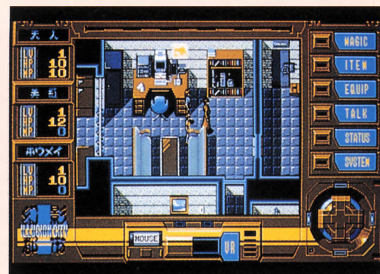
X68000用　5″2HD版　12,800円（税別）
イマジニア　☎03(3343)8911

幻影都市

突如，崩壊してしまった近未来の香港。原因は不可解な天変地異とされ，崩壊寸前に全世界へ発信された映像には魔物の姿が捉えられていた。

各国の調査活動も空しく，事態は謎に包まれたまま，時間だけが過ぎていく。そんななか，調査，および魔物の排除を請負おうという集団が現れた。国際情報企業集団SIVAである。SIVAはこの任務を遂行し，その見返りとして香港の統治権を得ることになる。しかし，そのSIVAの最高指導者“魔天王”は，魔天教という怪しげな宗教団体の教祖でもあった……。

こういう舞台設定のもと，主人公の天人は多数の個性的な仲間とパーティを組み，魔物，そして，SIVAに立ち向かっていく。敵側も個性派揃いでストーリーに幅をもたせている。なお，掲載した画面写真は開発中のもので，製品版ではメイン画面の周りに枠がつくようだ。

X68000用　3.5/5″2HD版7枚組　6,800円（税込）
ブラザー工業(TAKERU)　☎052(824)2493

CGAマガジン創刊号

少し発売が遅れたり，2枚組に増えてしまったりというトラブルはあったものの，すでにCGAマガジンは発売されている。内容を詳しく紹介しておこう。

まずは創刊記念として，かまたゆたか，マリオ古本，マクラーレン藤井の3人によって制作されたF1のアニメーション11カット。これは本誌でも掲載されているが，熾烈なコーナーリング，直線を突っ走るベネトンのフォロー，眼下を通過する，カメラに突っ込んで来る，などといった迫力のあるカットが揃っている。

そして，投稿作品も気合いが入っている。有川キラーさん（東京都）の“オリジナルカラーリングF1”はズームアップ2連カットとランニング3連カット，平山敏明さん（栃木県）の“直列4気筒エンジン”は直列4気筒エンジンのシリンダーの動きの精密なアニメーションを収録している。ほかにも，大石俊雄さん（宮城県）の超精密“零戦・96式陸攻”，河野悦昌さん（徳島県）の“オリジナル戦闘機”，石川龍さん（愛知県）の“F117ステルス戦闘機”とすごいデータを満載。

さらに，今回のDōGA CGAアニメーション講座の連載でも触れられているとおり，「CADを使わせたら日本一」のチーム“TOSAKA”のあ〜る田中氏による，F14戦闘機も入っている。

これらのアニメーションに使用している主な形状データはデータベースにまとめられているので，必要に応じて展開し，自分の作品制作に利用することができる。

プログラムは新作のタイムチャートエディタTCHED.X，疑似モーションブラーMOB.X，バージョンアップされたREND.X，RENDXVI.X，BGMAKE.X，FF.X，BETA.X，FFE.X，KAMA.X，EPA2.X，IC.X，SCROL.Xが用意されている。他機種のプログラムも予告どおり収録。

X68000用　3.5/5″2HD版2枚組　1,600円（税込）
ブラザー工業(TAKERU)　☎052(824)2493

1992年12月の月間売り上げベスト10

POINT	タイトル	発売元	発売日
826	オーバーテイク	ズーム	'92/11/20
682	テラクレスタ/ムーンクレスタ	電波新聞社	'92/11/20
646	ストライダー飛竜	カプコン	'92/11/27
413	ロードス島戦記II	ハミングバード	'92/11/20
305	スクウェア・リゾート	ファミリーソフト	'92/11/20
215	雀JAKA雀	エルフ	'92/12/24
197	MATIER	サンワード	'92/10/9
143	キングス・ダンジョン	ソフトプラン	'92/11/25
125	Communication SX-68K	シャープ	'92/11/25
98	キャノンサイト	日コン連企画	'92/7/4

TREND ANALYSIS

新作ソフトのタイトル数の少なさから予想はしていたが，ほとんど前回と同じような結果になった。特に1位から4位まではまったく同じである。

発売当初の熱狂ぶりからすると，「オーバーテイク」はやや落ちついた感があるが，それでも1位である。アンケートハガキの評価を見ていると，概ね高い評価を受けているようだが，なかには不満の声もちらほらと見受けられる。これはだいたいにおいては“期待が大きすぎた”というものであって，根本的には低い評価を下してはいない。ただ，それぞれ計ったように，まったく同じところの不出来を指摘している点が気にかかるのは確かである。

買った人の細かい評価に関しては，次号の“AFTER REVIEW”のページを参照していただきたい。とはいえ，たいていの人が買ったか，あるいはどこかでゲームを見ているだろうから，評価に対してあまり新鮮味はないかもしれない。

電波新聞社の“ビデオゲーム・アンソロジー”の第1弾，「テラクレスタ/ムーンクレスタ」も前回に引き続き2位，と健闘している。このシリーズはいい出足となったようである。第2弾も「チェルノブ」というシブい選択で，今後がますます期待できそうだ。

6位の「雀JAKA雀」はいわゆる脱衣麻雀というやつであるが，ふつうの4人麻雀ができたり，RPG要素が加わっていたりする。で，成人向きのものとそうでないバージョンが同時発売されている。どちらが売れているのかは謎だが，やはり成人向けのほうが売れているのだろう（これは単なる個人的見解である）。

この「雀JAKA雀」と，「キングス・ダンジョン」「Communication SX-68K」が今月の初登場ソフトということになる。

「キングス・ダンジョン」は内容的にはなかなかユニークなゲームであるが，発売前の知名度の低さと，ぱっと見の悪さが災いし，8位に納まったようだ。本誌のレビューも，発売から少しタイムラグができてしまったし。

9位の「Communication SX-68K」はSX-WINDOW用の通信ソフト。通信ソフトはいままでにもいろいろとあったが，SX-WINDOWをメインに使っている人には“待ちに待った”という製品であろう。

この“TREND ANALYSIS”では，約1年ほど，協力店からのデータ提供により，ソフト売り上げのベスト10を発表してきたが，諸般の事情により，こうしたかたちでの売り上げ順位発表は今月号で終わりということになった。協力していただいた各店の皆さんには心から感謝したい。

さて，問題は次回からの“TREND ANALYSIS”である。とりあえずアンケートハガキで集計をとる方向を検討しているが，どうなるかはまだわからない。送られてきたハガキを眺めて，考えることにしよう。

[データ集計協力店]（順不同）
九十九電機本店
OAシステムプラザ横浜店
ラオックス GAME館

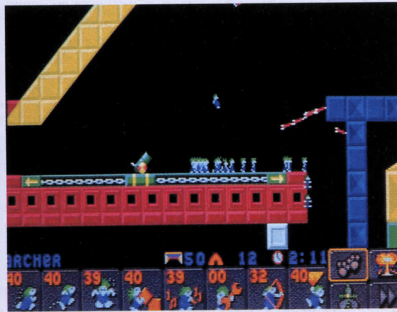
ウワサのソフトウェア（海外編）

Lemmings2-the Tribes

「レミングス」は1991年に発表されて，きわめて高い評価を受けたリアルタイムパズルアクションの傑作である。国内機種向けには1992年，イマジニアから発売されている。

ぞろぞろと行進を続け，放っておけば死地へなだれこんでゆくレミングたちに適切な命令を与え，安全な通路を確保させ，無事に彼らの家へ送り届けるのがプレイヤーの仕事。思考ゲームとしての要素とリアルタイム性とが高度にバランスをとっている。美しいグラフィックの中をちょこちょこと歩くレミングたちの動き，レミングの叫び声などの効果音，きわめつけはレミングを自爆させたときの花火大会と，演出も抜群で，一度遊んだらハマってしまう強烈に面白いゲームであった。

「Lemmings 2 - the Tribes」で，もっとも変わったのは，レミングの能力が数十種類に増えたというところ。「ポピュラス」と「ポピュラス2」の関係に似ていなくもない。従来の能力に加え，走る，跳ぶ，泳ぐ，といったように運動能力が強化され，風船を膨らませて空を飛んだり，弓矢を使うようになったり，はたまたスキーやスケートまでしたりと，使う小道具も増え，とても楽しそうである。技が多彩になった分，いろいろとユニークな戦略も立てられるようになっている。ほかにも，横だけでなく縦スクロールもするので舞台がより広くなったとか，早送りコマンドがついた（これはうれしい）とか，システム周りもいろいろと改良されている。

ゲームはといえば，前作の「救助率」という概念がなくなり，助けた数に応じたスコアが加算されるというシステムになった。これにより，面をクリアするために何匹を犠牲にして残りを救うか，などという悩み方をしなくなる。

デモバージョンの数面を遊んだだけだが，面白いことは面白いし，凝りように磨きがかかってもいる。巣箱の中のレミングたちを愛でるという要素が，より強くなっていることは間違いない。しかし，前作ほどにはゲームバランスはよくなさそうである。どちらかといえば，心よりも頭で感心した。この現時点での印象は，製品版まで保留しておくことにしよう。

あと，レミングが自爆する際に花火が見られず，けっこうつまらないものになっていた。もしもなんらかの理由（それも動物愛護とかの小賢しい理由）で廃止したのであれば，レミングスにおける重大な損失である。面によって自爆のしかたが違うようなので，取り越し苦労なんだろうけどね。 (A.T.)

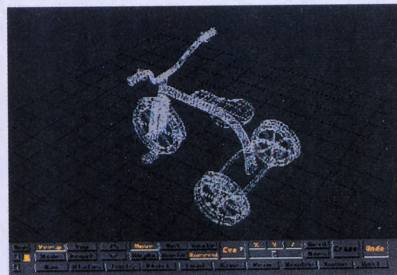
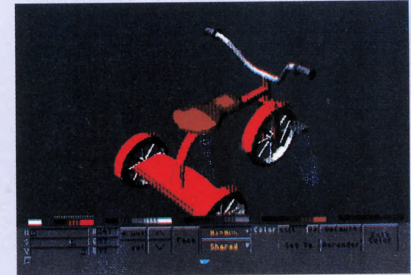
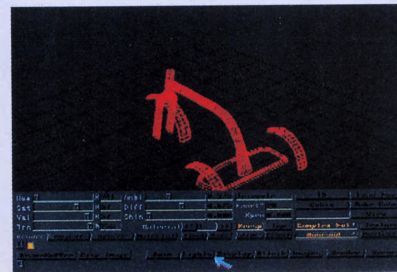
開発元　DMA Design

ウワサのソフトウェア（海外編）

Caligari2

ここでは本当はMacintoshの「Tree」というソフトを紹介しようと思ったのだが，入手できなかったので次回に譲ることにする。「Tree」は名前のとおり，木を自動的に描いてくれるソフトである。で，ほかをあたってみたのだが，めぼしいソフトが見つからなかったので，２月号の特集の囲みでちょっとだけ触れられていた，「Caligari2」を取り上げてみたい。

このソフトは，かなりユニークな機能を取り揃えたAMIGA用3Dレンダリングシステムである。まず，モデラ，シーンエディタの両方とも，基本的にパースペクティブビューですべての操作を行う，というところがいい。

マウスの操作に従って，オブジェクトの拡大/回転/移動といった動作がパースペクティブビューで，しかもリアルタイムに確認できるのは気持ちよく，かつ，わかりやすい。もちろん，きっちりと位置合わせをしたいときなどは，トップビュー/サイドビュー/フロントビューに切り替えることができる。遅い（普通の）マシンを使っている場合や，オブジェクトがあまりに複雑で計算速度が追いつかない場合には，ボックスに置き換えられて処理される。

次に目につくのはモデリング機能。このソフトのポイントエディットは実にユニークである。頂点をいじれるだけではなく，オブジェクトから線や面を切り出し，それに対して，通常のオブジェクトと同じような操作を行えるのだ。つまり，円柱の真ん中あたりを絞って（指定した面を縮小する），つづみのようなオブジェクトを作ったりすることができる。

面や線の切り出しは，ある辺上の任意の位置に点を追加し，その点の集合で線や面を構成させるという方法で行う。切り取った線や面をスライドさせて，新たな線や面とすることも可能。これは文章ではわかりにくいと思うが，切り出した断面（線）を平行移動させることができる，とでもいえばいいのだろうか。

このソフトは最近バージョンアップされて，「Caligari24」になっている。値段は399ドルだから，約60,000円というところか。

ともかく，自由に，しかも直感的にモデリングができるというのはスゴイことだ。

発売元　Octree Software

シューティング,ここに極まれり

Yaegaki Nachi
八重垣 那智

画面いっぱいにばらまかれた敵の弾を避ける。敵は出現した直後に叩く。危なくなったら必殺技のボンバー。「究極タイガー」にはシューティングゲームの快感が見事に凝縮されている。まさに“究極”というべきか。

ゲームが好きな人を理解するのに最も簡単な方法は，天気の話や食い物の話をすることではない。その人の好きなゲームや，思い出のゲームの名前を聞いてみることである。ごくごく平凡なミーハーゲームのタイトルが返ってくるかもしれないし，誰も知らないようなカルトなゲームが意外な人物から飛び出すかもしれない。それがどんな結果になろうと，そのゲームへの入れ込みぐあいや思い出が，その人を非常にわかりやすく表現していることに違いはないのである。

よく自分でこういう質問をするせいか，当然といえば当然のように相手からも同じ質問が返ってくる。そうしたときに私が答えるゲームは,「グラディウス」でもなく「ストリートファイターII」でもなく，今回X68000版の登場した「究極タイガー」であることが多い。

それは，単に好きだからだけではなく，ゲームセンターに登場する前のロケテスト段階から，いままでずっと遊びつづけてくることのできた縁の深いゲームだからである。究極の名にふさわしい，シューティングゲームの伝説の道を一緒に歩んできたといっても，過言ではないだろう。

それから4年の月日を経て，ついに自分のX68000で,このゲームが遊べるという話を耳にした。しかし,「飛翔鮫」の残念な移植や，PCエンジン版，メガドライブ版といった家庭用ゲームを見ると，つい不安に駆られてしまい，いっそのこと無理ならば出なければいいとまで思ったこともあった。そしていつの間にか1年が過ぎ，ある日サンプル版を見たときに，それは私の杞憂だったことに気がついた。そこには紛れもない「究極タイガー」があったのである。

究極への入口

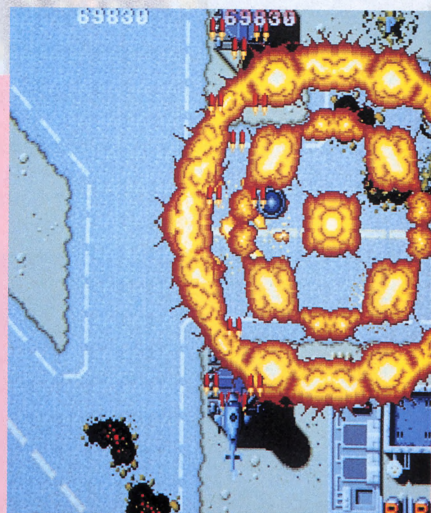
ここで「究極タイガー」とは，いったいどんなゲームなのか確認しておこう。このゲームは1987年の11月に発売された縦スクロールタイプのシューティングで，ヘリコプターを操作して，地上や空中の敵を撃破していくのが目的だ。単純明快な内容とひと筋縄ではクリアできない奥の深さを兼ね備えており，難しいながらもバランスの取れたゲームという評価が一般的である。当時はタイトーから発売されたが，この一連のシューティングを開発しているのが“東亜プラン”という会社であるという情報は，一部には知られつつあった。

操作は単純で，自機をレバーで操作し，2つのボタンでショットとボンバーを発射するようになっている。ショットは地上と空中の敵を同時に攻撃でき，ボンバーは回数が限られているが，その攻撃力と敵の弾を消す効果でピンチを救ってくれたりもする。こういうスタイルはいまでこそめずらしくないが，当時は王道といわれるシューティングゲームのスタイルはなかったため,「飛翔鮫」と並んで，この「究極タイガー」は非常に独特なタイプであったといえる。このテのゲームというのは，ここから始まったのである。

特定の敵を倒すとアイテムが出現し，それによって10段階のパワーアップと4種類の武器チェンジ，ボンバーの補給が可能になっている。これらのパワーアップで，より強大な敵を倒していかねばならない。不覚にも敵の弾に当たった場合は，自機とすべてのパワーアップを失い，残機がなくなったら，もちろんゲームオーバーだ。こう書くと，本当にどこにでもあるシューティングゲームのようである。

現在の定番ゲームの元祖であるわけだか

X68000用　5″2HD版2枚組　8,800円(税別)
KANEKO　☎0424(24)7752

特定の敵を倒すとアイテム出現

ボンバーの威力は絶大

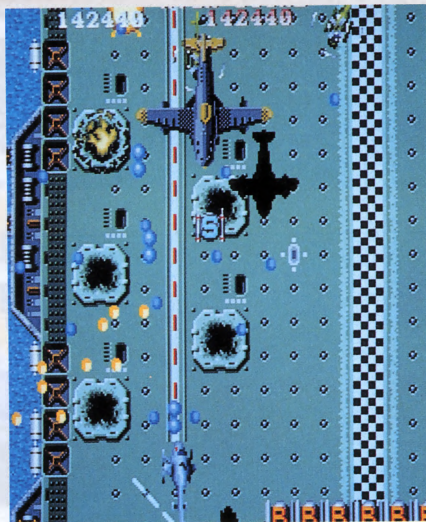
ら，馴染むことは難しくないだろう。複雑なパワーアップも，実際には非常にわかりやすく設定されている。それぞれ単一のアイテムに分かれているので，不要なアイテムを無視する必要があったり，取っているのにパワーアップしないというようなことは起こらない。すべての面でシンプルに洗練されており，美しささえ感じられるのは驚くべきことなのかもしれない。

ほかにも，「飛翔鮫」以上に念入りに描き込まれた背景や，独特のサウンド，“究極”という言葉の響きといった要素を含めて，このゲームに心を奪われた人は少なくないだろう。硬派なシューティングゲームでありながら幅広い支持を受け，アーケードゲームの業界紙で1988年のベストゲームに選ばれているのはハッタリではない。まさに，いままでの話題作に劣らないビッグタイトルゲームが，今回X68000に移植されたということになるのである。

虎との再会

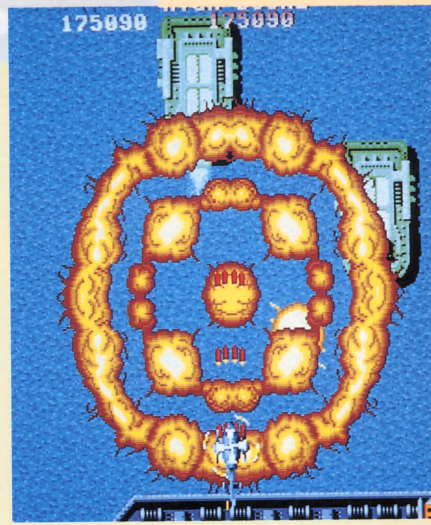
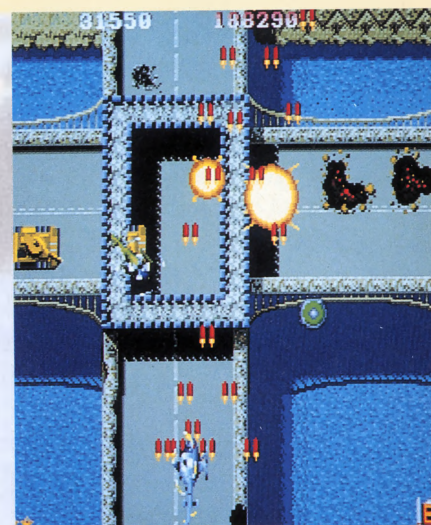
そこで，各界から不安の声の上がっていた移植の出来ぐあいだが，これには自信をもって，最も本物に近い移植であると断言してもいいだろう。家庭用ゲーム機のものが色褪せるほどの忠実さを実現しているのである。

グラフィックはそのままもってきたらしく，汚しの入った独特の質感を維持しているし，自機のパワーアップにも下手な省略や変更は加えられていない。敵の配置も忠実で，ボスの攻撃方法も同じである。まさに「そのまんま」の世界が目の前に広がっているのは，ちょっと感動的でさえある。

ゲームを始める前には，自機の機数や難易度を設定するためのモードがあるので，

アイテムを取りにいって死ぬことも多い

自信のない人は機数を増やしておくといいだろう。残念ながらイージーモードはないが，ショットの連射がサポートされているのはありがたい。連射は賛否両論あるが，あくまでも自分の好みで選択すればいい。

画面モードもここで一緒に設定できるが，デフォルトの画面幅の狭い設定のほうが本物の雰囲気が伝わってくるので，特に変える必要はないだろう。これも好みの範疇だが，やはりオリジナルに近いほうがなにかとよいと思われる。あと，しいていえば，ボタンの割り当てなども変更できるようにしてほしかったが，それほど不便には感じなかったので，あまり追及しないでおこう。

ゲームスタートはクレジットを入れてから行うようになっているが，最初に入る9クレジット分しか有効にならない。つまり，最初の1回を除いて，8回コンティニューが可能ということになる。このゲームにはエンディングがないので，この回数を多い

2隻まとめてボンバー攻撃

敵は片っ端から壊していこう

とも少ないともいえないが，どうせなら無制限にしてほしかったところではある。買ったからには，思いっきり遊びたいというのが人情だし，練習するにもコンティニューは重要なので，ちょっと残念な設定といえるだろう。とにかく準備ができたら，ゲームスタートだ。

究極の戦いの中へ

それでは面の紹介に移ろうと思う。攻略を詳しくやることはできないので，簡単な紹介の域を出ないかもしれないが，ニュアンスでも察してもらえればうれしいかぎりである。

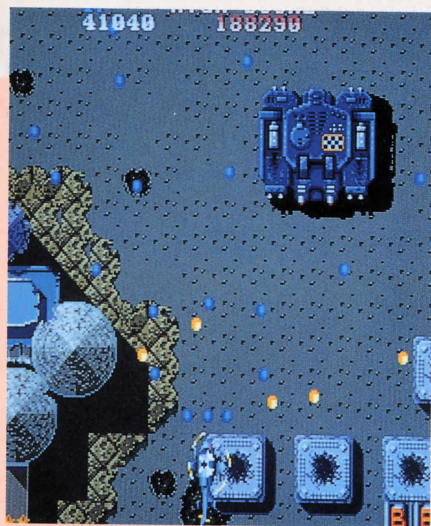
●ステージ1

主に荒れ地の上で戦車との戦い。常に画面を振り子のように大きく左右に動いて敵の弾をかわす，“切り返し”というテクニックをマスターしよう。これが，このゲームでは最も重要である。また，武器は原則的

面をクリアすると空母に着艦する

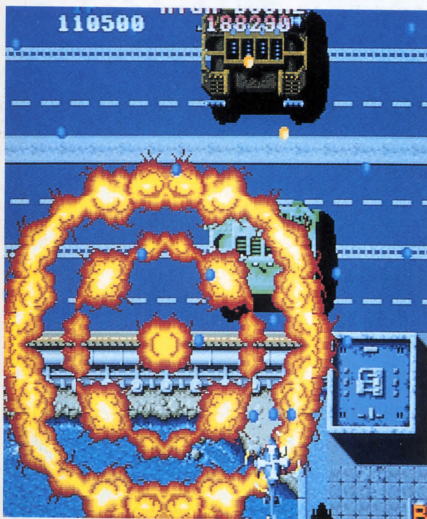
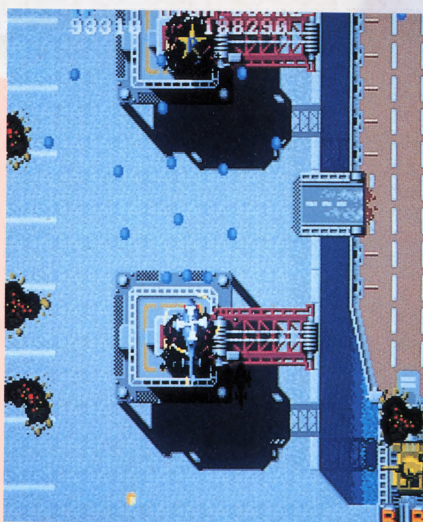
Bをとるとボンバー補充

青の武器は敵に当たりやすい

不意に後ろから射たれることもあるので注意

に青の5方向ショットを選ぶことが，基本である。好みで武器を選んでもいいが，このゲームはそんなに甘くないので，お遊び程度にしておくのが無難だろう。

後半は舗装された基地での戦いになる。戦車の配置を覚えて，不意を突かれないようにしよう。敵の配置を覚えるのも，このゲームではイロハのイだ。最後にはボスが待ち構えているが，2回ずつ発射される弾のリズムを覚えれば避けるのは簡単になる。逆サイドの戦車の弾には要注意。

●ステージ2

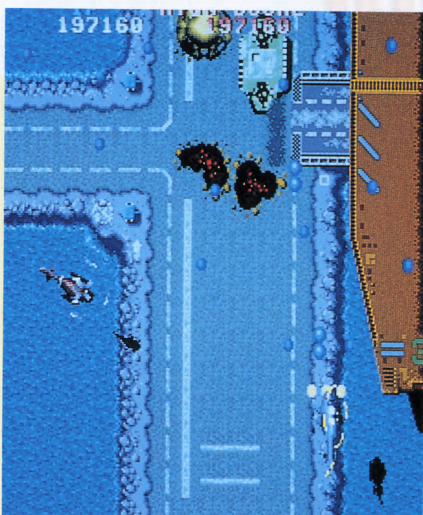
海の上から巨大な戦艦へと舞台が展開する。巨大戦艦までは，結構パワーアップが取れるので，前の面でのミスも結構取り返すことができる。巨大戦艦の上に現れる白い長距離爆撃機は，ボンバーを使って早めに倒したい。巨大戦艦の下から出てくる2台の四角いホバークラフト「アブラーゲ」にもボンバーを使うといい。ここにかぎらずボンバーは，転ばぬ先の杖のように先手先手で使うのがコツである。

ボスが弾を射っても，左右にちょっとズレるだけで避けられるので，あまり怖くないが，応援のヘリには注意。

●ステージ3

港での戦い。建物の陰から次々と戦車が現れるので確実に潰していく必要がある。雑魚ヘリの動きがいやらしいので，衝突しないように気をつけたい。基本を守ってプレイしよう。ボスは2台とも動き出すと手に負えなくなるので，画面に入るや否や手前の1台目をボンバーを併用して，速攻で倒すこと。

●ステージ4

海の面。小型艇がライバルだ。画面下から出てくるところは，あわてず確実に倒していこう。出てきたときに，真上に乗るようにして攻撃すれば問題ない。

後半上陸してからの攻撃も厳しいが，特にシャッター付きの砲台を射ちもらさないように注意すること。ボスは1台目が最初に迫ってきたときにボンバーを使えば，かなり楽に倒すことができる。

●ステージ5

弾の見えにくい海岸から草原を越え，敵の中間要塞を目指す。しかし，慣れないうちはあっという間にミスしてしまい，パワーアップできずにズルズルと終わってしまうのは，このあたりから共通の落とし穴である。地道に数をこなすことでミスを減らし，リカバリーもこなしてほしい。

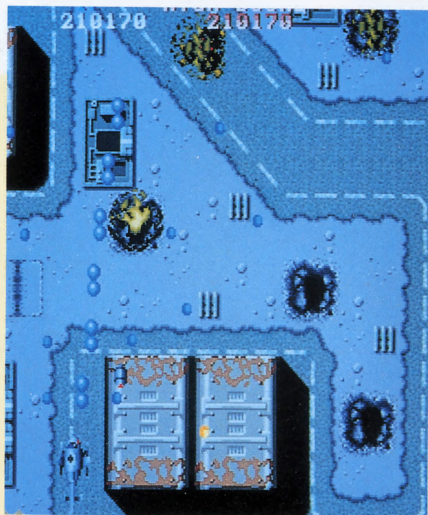
ボスは片側から斜めの弾で攻撃すること。ミスして青の武器にならなかった場合，かなりの危険性と難しさを伴う。そのためボスで死んだときは，再スタート直後にミスするとさらに前に戻されることを利用して，2つ前の草原地帯に戻してから再攻略したほうが確実だ。残機がもったいないが，これもこのゲームの重要なテクニックなのである。

このあとも全10面までゲームは続いていくが，それらはできたら自分の力で見てほしい。ここまでの5面を遥かに越えた地獄が待っているが，基本をしっかり押さえれば道は開けるように作られている。自力で攻略して，いつかは究極のプレイヤーになってもらいたいと思うのである。

色違いの虎

とまあ，ここまで手放しにほめてきたのだが，先ほど「本物に近い移植」と書いたように，これは完全なる「究極タイガー」ではない。ここでいちばん問題にしたいのはその解像度の差である。「究極タイガー」の場合，オリジナルの縦ドット数は320程度と思われるが，このX68000版では，縦ドッ

ボスはやっぱりボンバーで

敵ヘリもだんだん動きがいやらしくなってくる

シャッター付きの砲台は確実に狙おう

ト数は256である。この約60ドットの差をどうしているかというと，画面の上下をカットしているのである。これがどういう影響を与えるか，具体的に述べていくことにする。

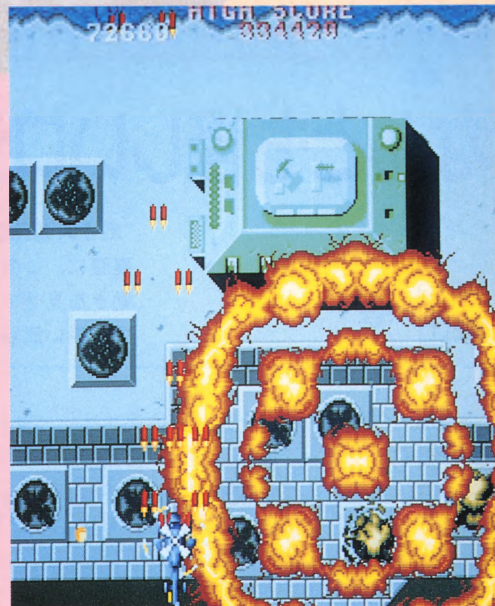
まずは画面が狭くなったことで発生するメリットから見てみよう。ひとつは，自機のショットが早く画面外に出ることで，連射の効率が上がるということ，それと一画面中に入る敵の数(特に戦車)が減るのでやさしくなる，というようなことが挙げられる。実際に画面上端では，弾を射たなくなる敵もいるので，確かに楽になっているのは事実だろう。

通常弾もパワーアップすれば多少使える

このボスは動かない

デメリットを見てみると，画面が狭くなって弾避けが厳しくなったといったものから，敵が出てくるタイミングが遅れていきなり攻撃されるといった，プレイの感覚を左右する部分を含んでいるのに注意したい。敵に攻撃させずに，先手必殺のプレイが要求されるこのゲームでは，敵の出現タイミングは大きな意味をもっているのである。

これらの違いはパターンに如実に反映されてしまうので，オリジナルで使えた攻略法や安全地帯などが役に立たないという事態が発生する。上達するほど，パターンプレイによって，敵を確実に安全に倒していく必要のあるゲームなので，これは非常に気になるところである。

特に，強力なボスに対して同じパターンが使えないというのは，攻略として非常に苦しいものがあるといえるだろう。そういった意味では，このX68000版には「似ていない」という判定を下すことも可能かもしれない。

ほかにも，左右の画面外処理や敵の当たり判定のサイズといったものから音楽まで，いくつか気になるものはあるが，やはり画面サイズのギャップはいかんともしがたいものがある。コナミの「グラディウスII」では安全地帯や特殊なパターンなどをわざわざ再現している。しかし，「究極タイガー」でそれをやった場合，ほかのバランスを崩す恐れも考えられるので，どちらがいいとはいえないのが難しいところではあるのだが。

究極の思い出

ここまで細かいことが気になるほど，基本的には移植が成功しているという見方も当然ながらあるわけで，X68000版の「究極タイガー」の出来はかなり完成度の高い部類に入るのは間違いない。独自の攻略を編み出すことを楽しみとして捉えれば，オリジナルよりもやさしい印象を受けるとはいえ，もともとの難易度の高さを考えると，オリジナル同様に長く楽しめるゲームであることには変わりはない。オリジナルでのハイスコアプレイヤーの目標は1000万点だったので，それを目指すのもいいだろう。

最後に注文をつけるとすれば，ゲームオーバー時の処理であろうか。起動画面までそっくりに作ってあるのだったら，きちんとネーム入れや，デモプレイなども移植してもらいたいところである。

ネーム入れで流れる背景を見ながら，不思議な満足感にひたれるところまで移植してあれば，「究極タイガー」をプレイしていた頃の思い出が帰ってくるような気がして，ちょっともったいないような，寂しいような気分になった。

このゲームが好きになったすべての人のために，ほんの小さなこだわりがあれば，このX68000版はもっともっと愛着のわくものになったのかもしれない。

死んだあとはやっぱり武器が寂しい

究極の移植を求めて

初めてゲームセンターでこのゲームをプレイしたとき，あまりの難しさに1分ほどでゲームオーバーになったような記憶がある。その当時の手帳を見るといろいろとメモが残っているので，いま思い出す以上のインパクトがあったのかもしれない。それほど印象深いゲームであったことだけは間違いないので，こうやって長く遊びつづけてくることもできたのだろう。

本文ではあまり書かなかったが，サウンドは「飛翔鮫」のほうがいい出来になっている。それでも，そういったことを忘れてプレイできるのは，移植全体のクオリティが高いからだろう。オリジナルの雰囲気が大事にされているのはうれしいことではないかと思うのである。しかし，何度やっても，ついオリジナルのつもりで敵を狙いにいって返り討ちにあうのは，移植のせいじゃなくて，私に学習能力がないからなんだろうな。人間は生涯学習だな，うんうん。

総合評価	0　5　10
ゲーム性	★★★★★★★★★
技術	★★★★★★★★
サウンド	★★★★★
グラフィック	★★★★★★★★
熱中度	★★★★★★★★★
マウスモード(笑)	★★★★★★★★★

なんとも怪しいロシアンヒーロー

Shibata Atsushi
柴田 淳

“戦う人間発電所”。一種異様ともいえるサブタイトルとは裏腹に，このゲームの主人公「チェルノブ」は実に華麗な動きを見せてくれる。細やかな走り，軽快な宙返りのなかでの鋭い武器さばき。一連の美しい動きで敵を倒せ。

宇宙線，すなわち宇宙を飛び交う高エネルギーの粒子のうち，地球に向かうものの大部分は，大気に妨げられ，地上に達することはほとんどない。ところがたまに非常に鋭い一線というのがあるらしく，単位面積あたり，つまり1cm²あたり毎分1個の割合で地上に降り注ぐのだそうだ。

すると当然，僕たちの体は日々相当量の宇宙線に貫かれているのだ。こうして雑誌なんか読んでいるあなたの体にも，今数個の放射線がグサと刺さったはずである。そのうちのさらにいくつかは，運よくあなたのDNAのそばを通りすぎ，遺伝子情報を書き換えてしまったかもしれない。

で，なんかの拍子に遺伝子がうまいぐあいに書き換わって，人体大変革が起こらないものかと，僕はずっと期待しているのだ。たとえば目から熱線が出るようになったり，人差し指からレーザーが出るようになったりしたら素敵だ。でも，もしそうなったらエネルギーがいっぱい必要だろうから，いらなそうな臓器，いっそ胆嚢なんかが核融合炉になってくれたら好都合だな。あるいはもっと現実的に，肩甲骨がニョキニョキッとのびて羽になるとか，奥歯を押すとマッハで突っ走れるとか。

しかし，そんなふうにDNAが書き換わる前に，ガンか白血病にでもかかってるか。現実はキビシイよな，まったく。

そんな世間の空っ風に吹かれた日には，せいぜい「チェルノブ」でもやって，変身願望を慰めずにはいられない僕なのだ。

それがチェルノブだっ

読者のうちで，いったいどれくらいの人がオリジナルの「チェルノブ」をプレイしたことがあるのだろうか。少なくとも，あまり有名なゲームではないような気がする。でも，だからといってつまらないかというと決してそうではない。あのマイコンソフトが移植するくらいだから，やはり名作のひとつに数えられるアクションシューティングなのである。

知らない人のためにこのゲームをひと言で表現すると，ヤバイ「魔界村」とでもなるだろう。だいたい設定からしてヤバイのだ。なんでもロシアの炭坑夫が原発事故で放射能を浴びてしまい，超能力を得る結果となった。前述の人体大変革を地でいっているのである。そして当然出てくる悪の秘密結社。あとのストーリーは，あまりに恥ずかしくて僕には書けない。

写真を見ればわかると思うけど，絵柄もけっこう独特である。主人公の「チェルノブ」がまたやたらと滑らかに走る。

ところで，こういうヤバめのゲームというのは，ともするとその独特さゆえに敬遠されがちだ。だけど考えてみると，独特であることを維持するためには，ゲーム全体が独自の世界観みたいなもので貫かれていなければならない。だからおざなりにかっこいいゲームなんかより，この「チェルノブ」のように異色なゲームのほうが，作る側にしてみればよっぽど気が抜けない。少しでも気を抜くと，そこだけ統一性が抜け落ちてしまって，ついには全体的に台なしになってしまうのである。

つまりだ，こういうヤバイゲームをゲームとして成り立たせるためにはたいへんな苦労がいるわけで，それからさらに名作として抜きん出るということは，これはものすごいことなのではないだろうか。

パワーアップの3大要素

さて，名作名作と連呼しても，このゲームを知らない人にはどこが名作なんだかわからないだろうから，そこらへんも少し掘り下げてみる。

まずこのゲームは，アクションシューティングであるクセに，なんと画面が強制スクロールする。おまけに，「チェルノブ」の後退速度はスクロール速度に等しいので，いったん進めば戻れないという，男気のあるプレイが求められる。これは考えるとたいへんなことで，たとえばこのゲームでは，火を打ち消すとパワーアップアイテムが落ちてくるのだが，うっかり通りすぎてしまうと，そのアイテムはもうあきらめるしかないのである。

そのうえ，チェルノブ様はレバーを左に入れればそちらを向くといった，アクションゲームにありがちな軟弱なことはなさら

X68000用　5″2HD版　4,800円(税別)
電波新聞社　☎03(3445)6111

怪しい手がアナタを怪しい世界へ誘う

犬みたいだけど，本当は首長竜

ない。「向きを変えたいなら，ボタンを押せ！」というのが，彼のモットーらしい。

ではここで，算数のできるあなたは声を出して数をかぞえてみよう。ショット，ジャンプ，方向転換。そうだ，このゲームは５年前のゲームでありながら，３ボタンというなかば破綻的（？）な操作系を導入しているのである。しかも３つとも頻繁に使うボタンなので，初心者はまずこの操作系に慣れるのに苦労する。

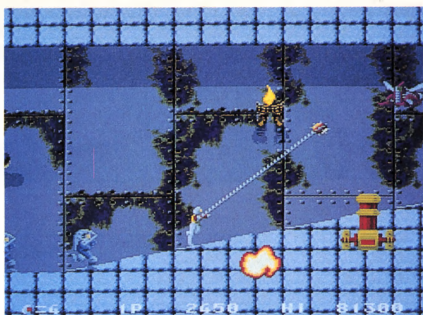
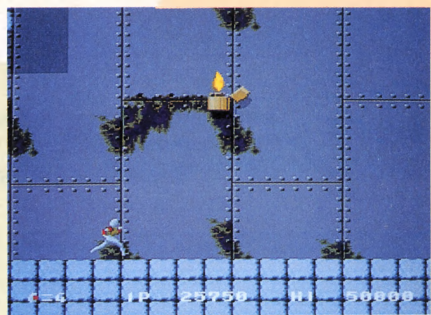
でも，そんなマイナス要因を押し退けてまでプレイ意欲をそそるものを，このゲームはもっている。ひとつには先ほど挙げた独自の世界観があるのだが，そのほかで特に紹介したいのが，簡単明瞭かつ要点をついたパワーアップシステムである。

まず，「チェルノブ」には５種類の武器が用意されている。自動追尾ミサイルあり，強力電撃ムチありの武器のほかに，それぞれの効率を高めるアイテムが別に３種類用意されている。

シューティングゲームをするプレイヤーにとって，武器はどのようにパワーアップしていってほしいか，つまりより楽にゲームをプレイするためにはどうなればよいかを考えよう。まず最初に思いつくのが，武器の威力を高めるという方向だろう。１回ボタンを押すごとに敵に与えるダメージが増えるので，プレイヤーはその分楽になるわけだ。

つぎに考えるのが，弾の連射数を増やすということだろうか。たいていのシューティングゲームというのは画面に現れる弾数が決まっており，その数だけ弾が画面にあれば，いくら連射しても新しく打ち出されることはない。が，連射数が高まりさえすれば，絶え間なく弾幕を張れるのだから，これまたプレイヤーは楽になる。

３番目の要素を挙げるには，特殊な制限を設けなければならない。たとえば，弾の届く範囲が狭められていたり，または速度が遅めだったりする場合だ。前者はより遠くに弾が届くようになればありがたいし，後者の場合はより速く弾が飛べばよい，ということになる。あとのほうは，さらに２番目の要素とも関係がある。つまり，弾が速くなれば，それだけ早く画面から消えるのである。

「チェルノブ」のパワーアップシステムは，まさにこの３点を押さえている。そしてこれだけ明確なシステムを打ち出したゲームというのは，案外少ないように思える。というよりここまでパワーアップを明確にしてしまうと，逆にバランスを取るのが難しくなるだろうから，誰も手を出さない，というのが本当のところかもしれない。

使える電撃ムチ。当たり判定が広い

火のついたライターには高得点アイテムが

わかっていただけただろうか。設定が独特という裏に，しっかりと地に根を下ろしたシステムをもったゲームが，この「チェルノブ」なのである。５年前のゲームなのになぜか新鮮さを感じる。変だからといって敬遠するのもいいが，もしオリジナルをプレイしていないならなおさら，結局ソンをするのはあなたである。

貫徹の完全移植

難点をひとつ。繰り返すと，このゲームは３ボタンの操作系を採用しているのだけど，X68000のジョイスティックというのは，２つのボタンしか認識できないような設計になっている。

じゃあ，どうするか。３つのボタンがあり，なおかつアタリ仕様であるメガドライブのパッドを使うための変換アダプタがついてくるのである。

メガドライブを持っていない人はキーボードで遊ぶか，それともジョイスティックのボタンを，A，B，A＋Bと使い分けるかの選択を強いられる。

結論を急ぐと，このゲームは３ボタンでないとちょっと遊びづらい。でも，移植版なのだから，移植先のハードの制限を受けるのはしかたないことなのかもしれない。それに遊びづらいといっても，ボタンの使用割り当てを変える機能が用意されているので，要は自分にあった設定を見つけ，あとはそれに慣れさえすれば，ボタン操作にとまどうということもないだろう。

移植という点でもうひとつ。「チェルノブ」では，武器を使って撃ち落とすほかに，ジャンプして踏みつけることでも敵にダメージを与えることができる。しかしオリジナルでは，傾いた地面ギリギリのところで踏みつけると「踏んだ」ではなく「当たった」と判定されるらしく，そこで１ミスという場面に悩まされたものだった。

地面に着いたかどうかというのは，地面のBGの１セル（８×８の格子のこと）に足が入ったかどうかで判定を行う。地形が傾いている場合，セルの上のほうがかなり空いてしまっているので，見た目は浮いていても，内部的には着地しているとして扱われるのである。

マイコンソフトが出すゲームというのはなにより完全移植が売りなのだが，このように高度なプログラミング方法に根差した部分まで，X68000版には受け継がれているのである。そのほか完璧さを挙げたらきりがない。

作り手側がこのゲームに注ぐ愛情と，完璧さを求めるこだわりが非常によくマッチしていて好感がもてる。かなりめざとく，プログラミングにも造詣の深いテストプレイヤーが，開発に携わったのだろう。

ぶっ，「ぶたさん」も頼むっ！

ささやかなアドバイスを。基本的にジャンプは垂直に。宙返りしていると，踏みつけの判定がなされないからである。

無限コンティニューがあるから，根性さえあればエンディングは必ず見られる。ただし，５面と最終面のボスはちょっとつらいかも。５面の鳥は，画面右端でジャンプしながら，羽と首の間から心臓を狙う。最終面は赤城山ミサイルを取って，適当にダメージを与えつつ定期的に頭を踏みつける。するとボスは左のほうに引っ込むので，それをひたすら繰り返す。

この「ビデオゲーム・アンソロジー」シリーズは，隔月で発売される予定とのことだが，このように過去の名作に光を当てるのは，僕は非常に尊い仕事だと思う。しかも，アンケートハガキに今後移植を希望するゲームを書けば，受け入れられるかもしれないという。ユーザー側にも，選択の道が開かれているのである。だから臆せずハガキを出そう。個人的には，「ぶたさん」あたりがいいと思うのだが。

総合評価

	0　　5　　10
ヤバイ世界観	★★★★★★★★★★
移植度	★★★★★★★★★
音楽	★★★★★★★
お買い得度	★★★★★★★★

ナンデモアリアリの蟻ゲーム

「SIM」シリーズにアリの生態をシミュレートするゲームが登場した。名前はもちろん「シムアント」。オリジナル制作はアメリカのソフト会社MAXIS，移植はいわずとしれた海外ゲーム移植の業師，イマジニアだ。

Nishikawa Zenji
西川 善司

私は小学生の頃，昆虫採集が大好きだった。夏休みは毎朝午前3時に起きて，ひとりで近くの雑木林へ自転車に乗って出かけていくという熱中ぶり。自分だけの秘密の木を蹴とばすと，ガサガサとカブトムシやクワガタなんかが落ちてきて，その音に歓喜の声を上げたものだ。

私の「昆虫好き」はこういった人気虫が発端で，以後とどまることを知らなかった。怪しげな「昆虫教室」とかいう昆虫のことを教えてくれる塾みたいなものにも通い出し，どこかに出かけるときは必ず昆虫図鑑を持っていった。だから部屋の本棚には土のついた昆虫関係の本がドサドサあった。

毛虫を楽しげに観察したり，ホテルの電光看板の下で蛾の死骸を集める姿は，大人の目にどう映っていたのだろう。

そこに蟻アリき！

多くの昆虫は単独で行動する。なかには団体で行動するものもあるが，アリほど発達した「社会的昆虫」はそうはいない。各アリにそれぞれ役割分担がなされ，その仕事を種族繁栄のために繰り返す。働きアリは食物を探しに出かけて巣に持ち帰り，幼虫や女王アリの面倒を見る。兵隊アリは縄張りを守るために侵入者を嚙み殺し，女王アリは卵を産みつづける。そして，羽アリは新たな領土を目指し旅立つ。

アリの社会的統制力と集団知性は人間に肉薄したものがある。アブラムシを家畜として飼育したり，敵種族の巣を攻撃，侵略して敵の卵を略奪，それらを自分たちの巣で孵化させて奴隷アリとしてこき使うといった恐ろしい習性を持ったアリまでいるらしい。兵隊アリの中には敵のもつフェロモン（匂い）と同じものを出して，敵をだまし混乱させるスパイがいるとか，働きアリは一生のうちに何度も転職を繰り返すとか，なんだか聞いていてゾッとするほど，アリは高等な昆虫なのだ。

こんなゲームアリ？

「シムアント」には大きく分けて，4つのゲームモードがある。ひとつは「入門編」。これはゲームの概要や操作方法をプレイヤーに教授してくれる親切なモードだ。画面に出てくるメッセージに従って行動して，最後までプレイすれば「シムアント」の面白さがわかってくるはずだ。

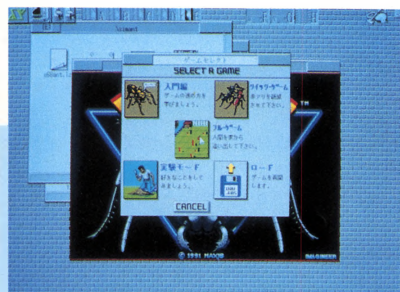
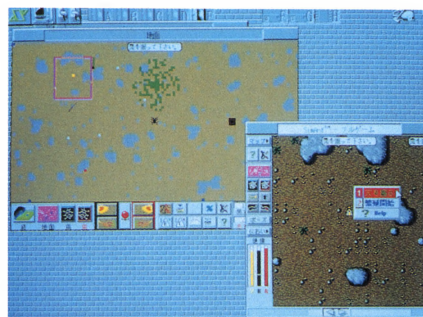
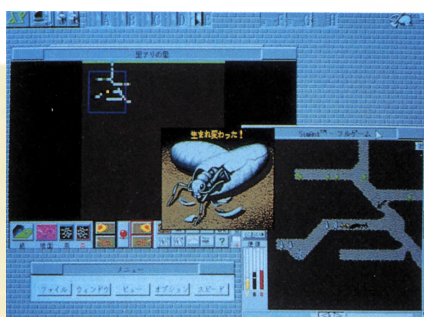
2つ目は今回メインで紹介する「クイックゲーム」。これは庭の一画での赤アリと黒アリの勢力紛争を描いたものだ。自分は黒アリのリーダーとなり仲間を増やして統率し，赤アリの女王を殺すのが目的だ。入門編をクリアしたら，このモードをプレイしてみよう。

3つ目は来月詳しく紹介する「フルゲーム」。これは，人間の住む「庭つき1戸建て住宅」の庭の一画に降り立った1匹の黒アリの女王が，自分の種族を繁栄させて先住の赤アリと人間を追い出し，この家1軒を占拠するのが目的という（アリにとっては）壮大なテーマのゲームだ。「クイックゲーム」の要素に加えて，領土拡張の戦略の要素までもが絡んで，「シムアント」がいっそう面白く感じてくるモードだ。

4つ目は「実験用ゲーム」。マップを歩き回るアリの巣に対して，巣を埋めちゃったり，クモを置いて意地悪したり，殺虫剤をかけたりする。子供の頃にアリの巣にしたイタズラを，画面中のアリに対しても行えるという科学教材的なゲームモードだ。これも来月紹介する。実はこのモードが本来の「SIM」シリーズのあるべき姿だという人もいる。ちなみに，このゲームモードのみ，勝敗という観念が存在しない。

それでは，今月の話題のメイン「クイックゲーム」の攻略に移ろう。途中「入門編」を終わらせていないとわからない表現もあるかもしれない。ご了承あれ。

愛の巣に幸せアリ！

まず，女王アリと2人きりの巣の中からゲームが始まる。ここで女王アリと「恋と愛の違い」について語り合ってもかまわないが，穴を掘り広げて巣を拡張し，女王アリの卵を産むスペースを確保してやることを奨励する。適当に巣を広げたら，外に出て食べ物を探しにいこう。「シムアント」では食べ物はマップ上で緑色のボールで表される。大軍勢で毛虫やクモなどに襲いかか

X68000用　5″2HD版　12,800円（税別）
イマジニア　☎03(3343)8911

フルゲームは羽アリが繁殖を始めるところから

死んでも生まれ変わるだけ

って勝利した場合，その死骸は「ブビ」とかいって食べ物，すなわち緑のボールへと変化する。うーん，大軍勢で死骸を運んだりする光景を見たかった私としては，このゲーム上のデフォルメは少し残念だ。

食べ物を拾って巣に帰ってくるまでの一連の行動は，「フェロモン」と呼ばれる分泌物によって道にマークされている（フェロモンマップで確認することができる)。以後，プレイヤーの仲間アリはこのマーカーをたどって食べ物を取りに向かう。プレイヤーのアリはこの食べ物と巣の間の道しるべを作る，重要な役割をもったアリなわけだ。そういうわけで，新しい食べ物を発見したら，巣の間を必ず2，3往復はするようにしよう。

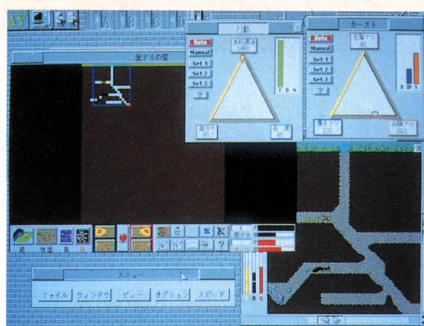
食べ物集めが軌道に乗ったら，今度は△Bのウィンドウを開いて,「食物収拾」「女王アリと卵の看護」「巣の拡張」の3つの行動パターンに適当な比率で味方アリの行動をコントロールしよう（序盤はそれぞれ80%，10%，10%くらいか)。

そして，次に△Cのウィンドウを開いて，出産させるアリの種類の比率を変更してみよう。生殖アリ（羽アリ）はクイックゲームにおいてはまったく無意味だ。この生殖アリはフルゲームのときにしか使わない。よって「クイックゲーム」では働きアリと兵隊アリしかいらないことになるのだが，食べ物が不十分なときは大食らいの兵隊アリの量産は破滅を誘う。序盤は働きアリ中心，後半は兵隊アリ中心の生産ライン(?)をオススメする。

敵アリ味方アリ！

地表にはクモという強敵が徘徊している。また，人間の足や芝刈り機といった（アリにとっては）突発的な災害の危険性もある。地表はアリにとって夢と冒険あふれるデンジャラスゾーンなのだ。

さて，地表にはいろんな虫たちがいるが，そのなかでも毛虫は攻撃してこないので，アリにとっては単なる動く食べ物。ゲーム展開に余裕があるようなら，毛虫に襲いかかってみよう。毛虫の進路にプレイヤーのアリを持っていき，動きを止める。そして，「味方召集」指令を出そう。アッという間に味方アリがよってきて，毛虫を取り囲んでしまうだろう。少したつと毛虫は緑のボールに変身してしまう（……合掌)。

クモやアリ地獄も大勢で襲いかかれば倒せるが，多大な犠牲を払うことになる。ただ，1匹を囮にして，クモを赤アリの巣のほうへ移動させてしまうといった高等なワザもある。クモは使いようによっては，制御不可能な強力兵器にもなるのだった。

行動およびカーストコントロールウィンドウ

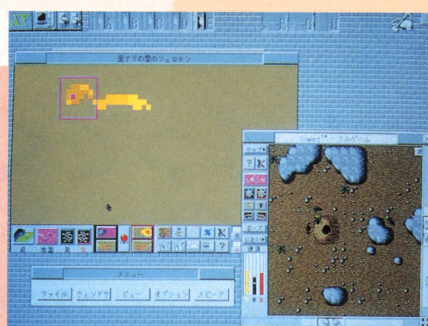
フェロモンを目で確認

戦うアリたちに栄光アリ！

自軍が優勢と思ったら，赤アリの巣へ攻め込もう。プレイヤーのアリが先頭となり味方を引率していくことになるので，プレイヤーのアリは戦闘は避けられない。そこで，プレイヤーの操作するアリを強い兵隊アリに「交替」しておこう。働きアリと兵隊アリは頭の大きさで区別できる。頭が大きくアゴが大きいのが兵隊アリだ。

さあ，何度も「味方召集」を実行して，いざ出陣。無残な犠牲者を出さぬために，遠回りになってもアリ地獄やクモのいないほうを通っていこう。

さて，まんまと赤アリの巣へ潜り込めたなら，そこでさらに，「味方召集」を行おう。直接自分で女王アリに挑んでもいいが，負けてしまうと生まれ変わってやりなおしなので，命を張った仕事は味方アリにさせるべきだ。赤アリの巣を黒アリだらけにできたなら勝利は目前だ。

敵の数を減らす目的のみで相手の巣へ攻め込むことも，もちろん重要。その場合には相手の巣の出入り口付近で「味方召集」をかけ，「警告フェロモン」を散布してしまおう。本来ならば巣を守るために使うフェロモンなのだが，これを敵陣地で使ってしまおうというのだ。召集された味方アリはフェロモンを散布した場所から離れなくなる。集まった味方アリは敵陣にもかかわらずその場所を動かなくなるのだ。つまり，彼らは敵と戦って死ぬまでその場所で戦闘を繰り返す「アリ機雷」になるのだった。残酷だが，勝つためには手段を選んではいけないのだ。ほーほっほっ（狂喜)。

“クイックゲーム”で存分に,「シムアント」のストラテジー（戦術）を修得してくれ。グッドラック！

X68000版は……

ゲームの難易度は低いと思う。なぜなら，私がプレイして2日目にしてフルゲームを制覇してしまったから。いまだに何に使うのかわからないウィンドウやコマンドもあるが大いに楽しめた。目新しいゲームだけに第一印象は「難解」かもしれないが，ふだんシミュレーションゲームをプレイしない人でも，必ず最後は勝てると思う。

しかし，X68000版は遅い。とにかく遅い。X68000がいくら（バリバリの486マシンやMacintoshと比べて）遅いといっても，ここまで遅いのにはうんざりする。SX-WINDOW用でなくて，アセンブラで専用に組み直せば（プログラム的に見て）この程度のゲームならば十分な速度のものが作れたと思うのだがどうだろうか。「ポピュラスII」があんなに速いだけに，今回の「シムアント」の遅さは納得がいかない。

理科の先生もアリがたがるゲーム!

それにしても西洋人の着眼点はスゴイ。だって，日本人だったら絶対作りそうにないジャンルでしょ，こんな昆虫の生態シミュレーションなんて。何重スクロールのシューティングゲームとか，RPGの戦闘モードに人工知能機能搭載とか，そんなので大騒ぎしてる我が国っていったい……。奇抜なアイデアだけでなくて，そういったアイデアをゲームとして完成させる企画力もスゴイ。そうだなあ，MAXISはまた生態シミュレーションゲームを作ってほしいなあ。猿の生態シミュレーションゲーム「シムモンキー」なんてのはどう？　語呂もいいし。猿山のボス争奪の身内争いから敵猿山の侵略問題とか，結構楽しそうなフィーチャー満載だと思うんだけど。あとは，ガン細胞になって人間の体を汚染破壊していく「シムキャンサー」なんてのは？ガンの怖さを知らしめる題材としても脚光を浴びるのでは。もし発売する場合は私にリベートちょうだいね。うひ。

総合評価

項目	0 ― 5 ― 10
ゲーム性	★★★★★★★★
独創性	★★★★★★★★★
処理速度	★★★★★
操作性	★★★★★★★★
グラフィック	★★★★★★★★
熱中度	★★★★★★★★★

がんばってくれい，俺の弾よ

Takahashi Tetushi
高橋 哲史

宇宙人の悪徳不動産グループに四角く切り取られ，リゾート地として持ち去られてしまった地球の土地。そんな理不尽な行動はもう許せない，と宇宙環境保護団体が立ち上がった。ホバー戦車「赤壱号」の戦いがいま始まる。

冬は寒いっ。3月号とはいっても，実際に本が発売されるのは2月18日なんだから，まだまだ寒いっ。しかし！　そんな寒さのなかにあって，萎えがちなゲーマーのファイティングスピリッツを熱く燃やすものがある。それはなにか？　思い出してほしい，初めて「ストリートファイターII」で他人と対戦したときのことを。あるいは「ポピュラス」での通信対戦，はたまた連爆ボンバーで黒こげにされたとき，君は熱く燃え上がるものを感じなかったか。そう，人間は人との戦いにおいてこそ，最高の情熱を燃やせるのだ。「人間VS人間」，冬はこれで熱くなるにかぎる。

ということで，今回ご紹介する「スクウェア・リゾート」も対戦でかなり熱くなれるゲームなのです。本当に熱すぎるので，ヤケドをしないようにご用心。余談ですが，最近は「ぷよぷよ」もかなり熱いです。メガドライブを持っている方は対戦しましょう。4連鎖地獄にはめてあげますから。

おや，どこかで見覚えが

さて，この「スクウェア・リゾート」は戦車対戦車の対戦ゲームです。ゲームを始めると，四角く区切られた“ポピュラス”状のフィールドに赤と青の2台の戦車。

ここで，「おや？　この画面はどこかで見た気が」と思われる方も多いでしょう。

X68000用　5"2HD版2枚組　4,500円(税別)
ファミリーソフト　☎03(3924)5727

そうです。実はこのゲーム，もとはアマチュアソフト(?)として世に出ており，すでに好評を博していたゲームなのです。それを，大幅に「リフレーッシュ！」したのが，この「スクウェア・リゾート」なのですね。ちなみに私は以前からこのゲームには結構ハマってまして，来る人来る人全員と対戦していた覚えがあります。始めると止まらないんですよ，これが。

戦車対戦車といっても，中身はちょっと変わっています。というのは，戦車がただの戦車ではないからです。どこがどう違うかといいますと，発射する弾がボーリング玉のように3Dのフィールド上をころころ転がるようにできているのです。遠くから敵を狙って撃つ場合は，まるでゴルフのパットみたいに「ラインを読む眼力」が必要になってきます。

さらにボタンを押しつづけることによっていわゆるタメ撃ちができ，弾の破壊力やスピードを細かく調節することができます。実際にプレイしてみるとわかりますが，この操作感覚がなんとも絶妙で，やっていくうちについつい引き込まれてしまう最大の要因になっています。戦車を操作するだけでも最初は大変だと思いますが，ぜひそこを乗り越えて楽しんでみてください。それだけの価値はあるゲームですから。

中野氏との死闘

さて，ここでちょうどOh!Xのマシン室にいた中野氏と対戦を試みることにします。中野氏は初めてのプレイ，私も今回パッケージソフトになった「スクウェア・リゾート」は，初めてプレイすることになります（ゲームの特徴を説明するために，フィクションの部分を加えています）。

私「中野さんは初めてやるんですよね？　ハンデつけましょうか」

中野「何いってるんですか。そんなものはいりませんよ」

おーっと，早くも発言に闘志が感じられるぞ。それでは，さっそくゲームを始めることにいたしましょう。舞台は“STAGE1 AREA1”。全体に傾いた地形で，特に突出した凸凹もなく，初心者の対戦にはもってこいの地形です。また，ちょうど真ん中にはドームのようなものが3つ並んでいますが，これは弾を当てると赤と青の弾を一定期間交互にまき散らす砲台になっています。あ，いい忘れましたが，自分の色の弾に当たった場合は，ダメージは受けないようになっているんですね。

私「それじゃ行きますよー！」

と，私はいうが早いか，坂を登って中野氏の上に回り，弾を連射！　発射された弾丸はいっせいに中野氏に降りかかります。

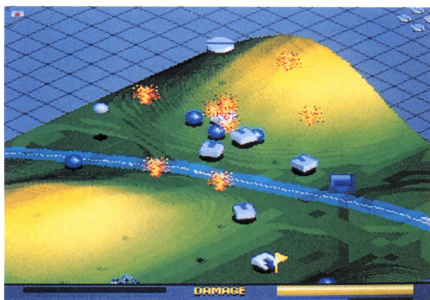
また途中で，前述の砲台に弾が当たり，円状に発射された弾も坂の下の中野戦車を襲います。

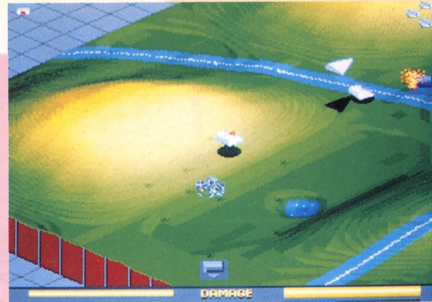
中野「ああ，ひってぇー!!」

早くも得意のセリフを連発しながら，それでも中野氏は懸命に弾をかわそうとします。しかし，いかんせんまだ戦車の操作に慣れていない中野氏。ボコボコに被弾し，煙を上げながらこちらの射程範囲から脱出していきます。しかし，私は非情です。逃げる中野氏をとにかく追いかけて，ダメージを加えます。

私「そおれ，それそれそれーっ！」

中野「あ，あ，あーっ！」

あえなく中野氏の戦車は爆発炎上。輸送機が新たに戦車を運んできて，フィールドの真ん中に置いていきます。中野氏の残機

袋だたきにあって，ボコボコに

はあと2機。
私「ふっふっふ～」
中野「……」
得意になった私は，また坂の上から中野氏を狙い撃ちます。またもや逃げまどう中野氏。これはもう楽勝かと思ったとき，信じられないことが起こりました。
どかんっ！
私「あっ……」
突如，私の戦車が爆発。調子にのって撃ちまくっていた私は，中野氏がとてつもなくでかい弾を「ためていた」ことに気がつかなかったのです。ためればためるほど弾の破壊力もスピードも上がりますから，坂の下からでも狙い撃ちができるのです。
中野「どうしました？」
私「……くぅー」
今度は輸送機が私の戦車を運んできます。が，輸送機がフィールドに私の戦車を置いた瞬間に……。
どかんっ！
私「ああぁあっ！」
なんと中野氏はあらかじめ輸送機が来る場所に狙いを定めて，弾をためていたのです。私の戦車は地上に降り立ってから，1秒もしないうちに爆発炎上。な，なんと，鬼のような作戦。これで私の残機はあともう1機。
私「中野さん，いまのはひどいですよお」
中野「避ければすむことです」
ぬおおぉぉ，上等だあ。私の最後の1機が地上に降り立ちます。今度は同じ失敗をしないように，すかさずジャンプ！　避けた自機の下を大玉送りみたいに巨大な弾が通りすぎていきます。やっぱり狙ってやがったなあぁ。
と，そのとき，ピロリンという音とともに，フィールド上に黄色い旗が現れます。
中野「何ですか，あれは？」
私「……」
私は黙ったまま，素早く旗を奪取。みるみる増えるこっちの体力。ふっふっふっ，これは体力回復のアイテムなのです。
私「いやあ，あれを取ると体力が回復するんですよ」
中野「……」
取ったあとでのしらじらしい説明に，さすがにむっとする中野氏。今度は朝日新聞のような旗が現れました。これは弾丸の威力を増すためのアイテムなのですが，中野氏は何も聞きません。とにかく出てくるものは取ったほうがよい，という認識ができたらしく，中野氏は無言で旗のほうに向かいます。当然，私もその旗を取りにいきます。2人で押し合いへし合いやってる途中で，またもや事態が急転しました。
ひゅー……，ぐしゃ。
フィールドの端のほうの旗を取ろうともみ合っている最中に，中野氏が誤ってフィールドから落ちてしまったのです。あわれ中野氏の戦車はフィールド外でぺちゃんこ。これで中野氏の残機もあと1機です。勝負が振り出しに戻ったかと思いきや，フィールドに降り立った中野氏の戦車がいきなり，どかんっ！
中野「あぁぁあ，ひってぇ！」
さっきのお返しとばかり，私も落下地点に狙いを定めていたのでした。ちゃんちゃん！
ということで，なんとか私は勝利を収め，古参の面目を保ったわけですが，これですむはずがありません。すでに中野氏はスタートボタンに手をかけています。
中野「……もう1回やりましょう」
私「何度やったって同じことですよ」
かくして熱い熱いバトルが数時間にわたって展開されることになるのでした。

黄色い旗は体力回復

戦闘区域から下に落ちるとグシャグシャ

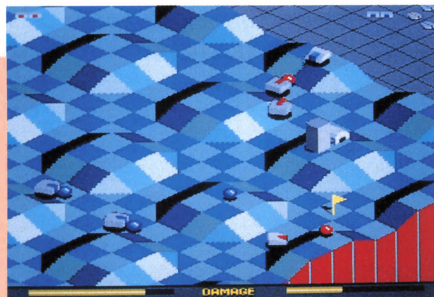
死ぬと飛行機が次の戦車を運んでくる

大きくなったのね

と，このように燃えるゲームなのですが，音楽やグラフィック，プログラムなどはどうでしょうか。結論からいえば，どれもほぼ合格点に近い仕上がりになっています。もとのソフトを知っているだけに，「うーん，見事に成長させたなあ」と感心してしまいました。
あ，それにともなって要2Mバイトになっていますので，増設していない方は注意してくださいね（最近ではほとんどの人が2Mバイト以上積んでいると思うけど）。
グラフィックもかなり描き込んでいますし，動きも気持ちいいです。10MHzでも十分遊べます（というか，16MHzだと速すぎてゲームにならない）。また，音楽もMIDI対応（SC-55，MT-32）でなかなかの聴きごたえです。ちょっと典型的なゲームミュージックっぽい気もするんですけど，そこがまたたまらない人にはたまらないんでしょうね。あ，ちなみにドライバはZ-MUSICを使用しています。
ということで，この「スクウェア・リゾート」はかなりお勧めです。ぜひ友達とプレイしてみましょう。室温が3度以上，上昇すること間違いなしです。

1人プレイは蛇足かな

記事の中では対戦のことだけしか触れませんでしたが，いちおうひとりでも遊べます。が，あまり面白くありません。というのも，むちゃくちゃ難易度が高いし，やっていて対戦ほど燃え上がるものがないからです。市販ソフトとして発売するにあたって，1人プレイをつけないわけにはいかなかったのでしょうが，やはりこのゲームのエッセンスは対戦にありますから，ちょっと蛇足のような気がしないでもないですね（逆にいえば，対戦モードだけでも4,500円分の価値は十分あります）。
できれば1人プレイよりもフィールドコンストラクションをつけてもらって，自分の作ったフィールドで対戦できると面白かったのではないでしょうか。3Dフィールドだから作るのはかなり苦労するとは思いますけど，次回のバージョンアップはそれで決まり（むちゃくちゃいってるな）。

総合評価	0　5　10
プログラム	★★★★★★★★
グラフィック	★★★★★★★
音楽	★★★★★★★
操作性	★★★★★★
熟練後の操作性	★★★★★★★★★

祝！X68030

待望のハードウェアとソフトウェアを追う

かねてからの噂どおり，ついに32ビットタイプのX68000シリーズが発表された。ここでは緊急レポートとしてもっとも気になる製品概要，性能，互換性の3点に焦点を絞って解説する。さらに詳しい解説は来月までお待ちいただきたい。

待望の新機種は，「X68030」だ。これまでの例から類推すると読み方は，おそらく「えっくすろくまんはっせんさんじゅう」が正しい。68030を略称で呼ぶ場合には「さんじゅう」ではなく「さんまる」，さらに通っぽく「まるさんまる」と読むのが望ましい。

ツインタワータイプ（正式にはマンハッタンシェイプタイプ）もコンパクトタイプもとりあえず同じ名前だ。

CPUには25MHzの68EC030が採用されている（「ならX68EC030じゃないか」というのは間違い。だってX68HC000じゃなかったでしょ）。メインメモリは標準で4Mバイト。増設メモリはXVI風の親子亀だ。

その他のハードウェア仕様は基本的に変わらず。一部で期待されていたような全面的モデルチェンジではなく，あくまでもX68000シリーズの最高峰モデルと位置づけられている。各種コネクタ関係はCompactと同じものが装備されている。

率直な使用感としてはあらゆる処理が軽い。もともとX68000のプログラムは10MHzで十分に動くものが揃っている。それが4，5倍のエンジンで駆動されるのだから当然か。68000は世間ではもはや高速CPUとは呼ばれていないが，そこらの32ビットパソコン顔負けのソフトが走っているというのは周知の事実である。パーソナルワークステーションの名はあながち伊達ではない。現在のところ，OSまわりを除いてX68030用に作成されたプログラムは存在しないが，この性能をフルに使ったアプリケーションがどんな水準になるのか……，なんとなくわくわくしてくるじゃないか。

68EC030とは？

まずは名前に冠せられたCPUから解説しよう。68EC030はモトローラの開発した32ビットCPU68030の姉妹版である。“EC”はEnbeded Contorolerの略でヨーロッパ共同体とは直接の関係はない。

もともとの68030が68020を高速化してM

X68030には，5インチFDDのツインタワータイプと3.5インチFDDのコンパクトタイプがある。いずれもチタンブラックカラーで，X68000シリーズならではの洗練された美しいデザインだ。そして赤いロゴバッジは，ひと目で最上級機とわかる高性能マシンのシンボルなのである。

表1　CPUベンチマーク結果

テストプログラム	X68000	V70	X68030
Dhrystone(μs)	1155.2 (1.00)	248.6 (4.65)	275.2 (4.20)
Whetstone(s)	17.27 (1.00)	0.68 (25.4)	4.47 (3.86)
Stanford nonfloat	567.5 (1.00)	115.1 (4.93)	110.7 (5.13)
Stanford float	2461.2 (1.00)	260.1 (9.46)	629.3 (3.91)
自己平方フラクタル	7:26:53 (1.00)	0:12:50 (34.8)	1:23:42 (5.34)

MU(メモリマネージメントユニット)をチップ上にまとめたものだったのだが，その68030からMMUをはずしてコンパクトにしたものが68EC030だと思っていい。性能は同じで低価格というモトローラの戦略商品である。組み込み制御装置のうち特に高性能を要求される分野で広く使用されている。ちなみに，コモドールの次世代を担うAMIGA1200では68EC020が採用されている。

モトローラにはもともと68851というMMUがあり68020で使われていたのだが，68030ではその簡易版が内蔵され，68EC030ではまた外され，68040ではさらに簡易化されたものが内蔵されるに至っている。

気になるのは，付けたり外されたりしているこの“MMU”というのはどの程度重要なものなのかだ。

MMUがないことのデメリットをひと言でわかりやすくいうと，概ね「UNIXを移植しにくい」ということになる。UNIXに限らず仮想記憶OSを動かすことが難しくなる。それ以外の用途ではあまり使われない。メモリ保護も甘くなるので「本当にちゃんとしたOS」も作りにくいのだが，うまくやらないとメモリアクセスが遅くなるのであまり喜ばれないかもしれない。とりあえず，現在のところHuman68kは仮想記憶に対応していないのであってもなくても関係ないのだが，将来的な展開を期待させるものがないとちょっとさみしい。

そ・こ・で，というわけでもないのだろうが，X68030ではなんと「68EC030が68030のソケットに入っている」のだ。将来的に仮想記憶OSが登場してきた場合でもCPUを差し替えるだけで簡単に対応可能だ。

ワークステーションへ向けて

68000の性能がDhrystoneでの評価ならV30と同程度というのはそう間違っていない。しかしX68000をPC-9801VMと比較してみると，できることや操作性の差は歴然としていた。80286というのは相当に速いCPUのはずなのだが，凄いアプリケーションというのは滅多にない。最近まで主流だった386にしても，ほとんどは16ビットCPUとしてしか使われていなかったのだ。

理屈では決して速いマシンではないMacintoshでもLCクラス(68020の16MHz：16ビットバス)でさえそこそこ快適にウィンドウを扱えるが，WINDOWSで同等の環境を得るためには少なくとも倍クロックの486が必要であろう。68000を使ったAMIGAと同等なゲームをPC互換機で実現するには4倍のクロックの386か486クラスのCPUが必要となる。

性能的にはインテル系のマシンはワークステーションクラスになってきているが，その環境は旧態然としたままである。なぜか？　誰もワークステーションのようには見ていないからだ。そしてMS-DOSの環境，WINDOWSの環境にあわせてしか用途を見いだせなくなっている。

モトローラ系のマシンが非常に長い製品寿命を持っているのに対して，2年間で陳腐化することが当然のように受け入れられている。強力なDOS/VマシンはPC-9801への当てつけのためにだけ話題にされ，ユーザーは高速なハードを持つこと自体が目的と化している。強力なマシンはWINDOWSのためだ。で，そのWINDOWSはAFTER DARKのために起動されている。

そう考えるとX68030には比較できるマシンがない。X68000ではワークステーションの環境を構築しようとする人たちがいた。アーケードゲームの完全再現を目指す人たちもいた。CGステーションを目指す人たちもいた。こういった天に向かって拳を振り上げるような行為が現在のX68000の独特な文化を築いてきたのだ。

いまユーザーに，ひと昔前のSun3と同じ性能を持ったマシンが提示された。かつてそれが完全に個人の占有で使われるようなことがどれだけあったろうか？　X68000は確かにアーケードゲームと同等のスペックだった。が，現在でも32ビットCPUのゲームというのは数少ないのだ。

その性能は？

68030はなぜ68000より高速なのかというと，

1)　基本的に命令が高速化されている
2)　32ビットアクセスが速い
3)　命令キャッシュがある
4)　データキャッシュがある
5)　拡張された命令がある
6)　アドレッシングモードが充実している
7)　パイプラインがきく

という理由を挙げることができる。もちろん，「高クロックで動作する製品がある」というのが決め手でもあるのだが，根本的に速いものは速いのだ。

基本となる最短命令実行時間はプロセッサの進化とともに短縮され，68000では4クロックだったものが，68010で3クロックに，68030で2クロックに，68040では1クロックになっている。

だからといって単純に68000の2倍速い

X68000新作ソフトウェア情報

今月のAFTER REVIEWはお休みさせていただきます

発売中のソフト

★機甲装神ヴァルカイザー　ブラザー工業(TAKERU)
X68000用　3.5/5″2HD版　4,800円(税込)
★ヴェルスナーグ戦乱　ファミリーソフト
X68000用　3.5/5″2HD版　9,800円(税別)
★チェルノブ　電波新聞社
X68000用　5″2HD版　4,900円(税別)
★沈黙の艦隊　ジー・エー・エム
X68000用　3.5/5″2HD版　12,800円(税別)

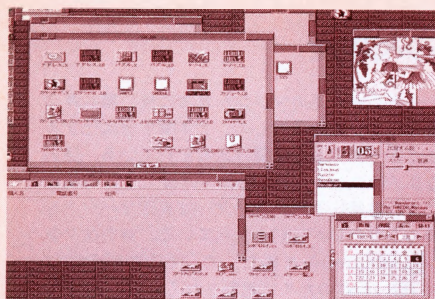
新作情報

★蒼き狼と白き牝鹿・元朝秘史　光栄　2/20
X68000用　5″2HD版　9,800円(税別)
★幻影都市　ブラザー工業(TAKERU)　2/28
X68000用　3.5/5″2HD版　6,800円(税込)
★ロボスポーツ　イマジニア
X68000用　5″2HD版　価格未定
★シムアント　イマジニア
X68000用　5″2HD版　12,800円(税別)
★メガロマニア　イマジニア
X68000用　5″2HD版　12,800円(税別)
★エトワールプリンセス　エグザクト
X68000用　3.5/5″2HD版　9,800円(税別)
★信長の野望・覇王伝　光栄
X68000用　5″2HD版　12,800円(税別)
★餓狼伝説　ホームデータ
X68000用　5″2HD版　8,500円(税別)
★Traum　M.N.M Software
X68000用　5″2HD版　価格未定
★鮫！　鮫！　鮫！　KANEKO
X68000用　5″2HD版　価格未定
★達人　KANEKO
X68000用　5″2HD版　価格未定
★エアバスター　KANEKO
X68000用　5″2HD版　価格未定
★サバッシュII　ポプコムソフト/グローディア
X68000用　5″2HD版　価格未定
★倉庫番リベンジ/ユーザー逆襲編　シンキングラビット
X68000用　5″2HD版　6,800円(税別)
★マージャンクエスト　SPS
X68000用　5″2HD版　価格未定
★麻雀悟空・天竺への道　シャノアール
X68000用　5″2HD版　9,800円(税別)

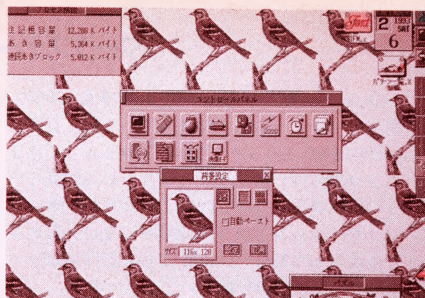
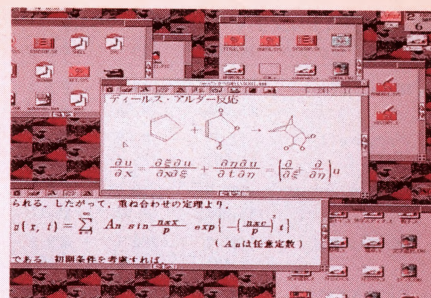
X68030を買うと，先着1000名の人にシャープから“SX-WINDOWデスクアクセサリ集”がプレゼントされる。スクリーンセーバー，アドレス帳，スケジューラ，電子手帳通信ツール，パズルなど12種類のアクセサリプログラムが収録されているそうだ。

SX-WINDOWデスクアクセサリ集

背景も自由に変えられる

最大の目玉はシャープペン.Xだ

わけではないので注意。統計的には68030の1命令あたりの実行時間は平均6.6クロックとなるそうだ。68030ではもう少し短いと思っていい。

●命令の拡張

ビットフィールド関係の命令が拡張されている。ビットフィールドとは，最大5バイトにまたがる32ビットまでの任意の大きさのビット列を扱うデータ型だ。グラフィックのRGB分離やテキストVRAMの処理には有効であろう。さらにこれまではnビットのシフトなどに6＋2nクロックのように時間が増えていたのだが，バレルシフタによって一度に処理されるようになった。どんなデータでも数クロックで処理される。

なお，従来からあった命令のうち，数値が一定の範囲内になるかどうかを調べる命令であるchkや条件つきtrap命令であるtrapvなどの使用はOSに禁止されているようだ。

●アドレッシングモードの拡充

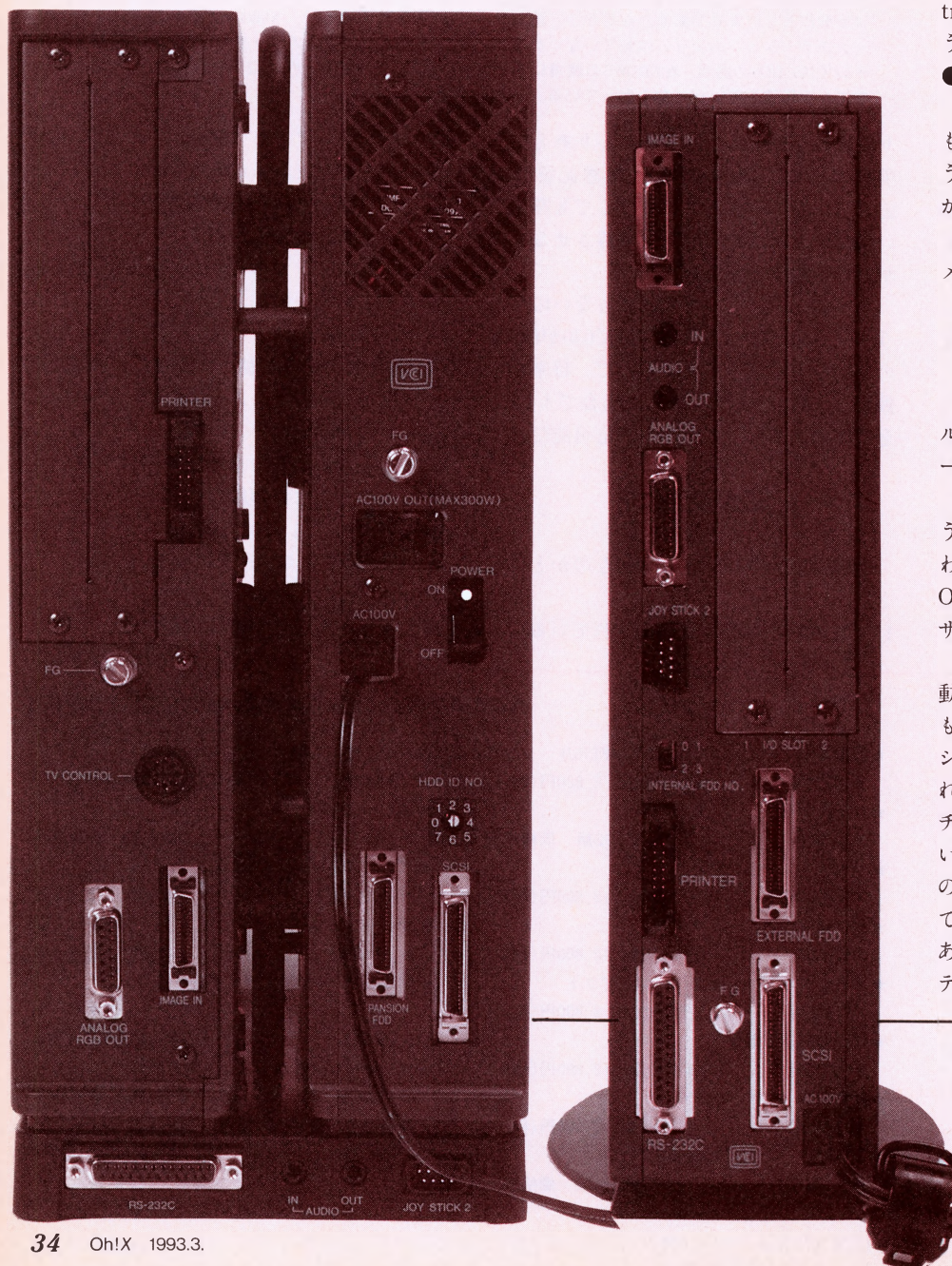
命令だけでなく，アドレッシングモードも拡充されている。従来は“d8”などのように表現されていたディスプレイスメントが16ビット，32ビットにも拡張された。“d16”，“d32”という具合だ。そのほか，メモリ間接などのモードが増えている。

互換性について

68030は68000に対してオブジェクトレベルでの完璧な互換性がある，ただしユーザープログラムにおいては。

冗談でも誇張でもなく68000系CPUというのは，もともとワークステーションに使われていたような石なので，ちゃんとしたOSが走るようにユーザーとスーパーバイザというものが厳然と分離されている。

ちゃんとしたOS上なら，マルチタスクで動作しているプロセスのひとつが暴走してもほかのプロセスは正常に動作できるし，システムが壊されるようなことはない。これくらいの信頼性が当たり前でないとマルチユーザーでのマルチタスクなどはできないのだ。ひと昔前の32ビットマシンというのは超高級機であり，個人で占有するなんてもったいないことは考えられないことであり，ちゃんとしたOSが載っており，システムが暴走なんかしないものだったのだ

数々のインタフェイスを標準装備したX68000シリーズだが，このX68030のリアパネルは意外とあっさりとまとまっている。X68030では5インチFDDモデルでもコンパクトタイプと同様，ハーフピッチのコネクタに統一された。また，立体視端子はツインタワーからもなくなった。

(するけど)。

しかし，ご存じのようにX68000というのはそれほど管理されたOS下で動作するわけではない。さすがにほかの16ビットマシンほどあっけなく飛んだりはしないが，性能をフルに発揮しようとするとみんなスーパーバイザ状態で突っ走ろうとする。これはアプリケーションではなくてOSの領域である。

これまでのX68000で動いていたプログラムは，相当ちゃんと作られたものか，相当手抜きをして作られたもの以外，OSの分野を含んでいるものだったのだ。こうなると上位プロセッサへの移行はスムースにいかなくなる。

アプリケーションでさえ互換性が危ないのだから，従来のOSであるHuman68kはそのままでは動くはずがない。これでは従来のIPL起動の市販ソフトは動かないことになってしまう。

そこで，今回のX68030では，XF３，４，５のキーを押しながら(それぞれ10/16/25MHzに対応) 起動することでROM Humanを起動し，フロッピーディスクから直接従来のソフトを使用できるように配慮されている。

Human68kを使っていないプログラムを立ち上げるときにはそのままリセットすればよい。運がよければちゃんと動く。直接起動できるゲームには，「ナイアス」や「トリトーンファイナル」などがある。X68000 XVIでうまく動かなかった「ナイアス」がX68030で完全に動作するというのも面白い。

現段階のメーカー発表値で約６割のソフトが動くとされている。動かないプログラムといっても実際に動かない命令はごく限られた部分にしかないと思われる。こういったものは簡単なパッチ当てでほとんどが動くようになるだろう。プログラムをチェックして個別の対応法が明らかになれば，Oh!Xで起動できないプログラムへなんらかの対応を行うことも考えている。

こういったディスクから直接起動するタイプのプログラム以外にもCPUを換えることでいろいろ引っかかりそうな問題はあるのだが，たいていのプログラムがそのまま動いている。ハードウェア側でかなりの問題を吸収しているようだ。X68000には，ハードウェアを直接制御するプログラムが特に多い。もし基本ハードウェアの変更があったなら，互換性はほとんど望めなかったのかもしれない。

新しいシステムソフトウェア

今回大きく変わったソフトウェアは，基本OS Human68k, SX-WINDOWシステムおよびアプリケーション，日本語変換フロントエンドプロセッサASK68K (AsK３) の３つである。

●Human68kver.3.0

Human68k ver.3.0はX68000とX68030で共用できるOSである。従来のソフトウェア資産を最大限に生かすことを重視したものになっている。いろいろ新機能も増えていると思われるが，現時点では，全貌は見えていない 。

IOCS ROMはもちろん32ビット対応になっているが，それ以外にも2DDディスク(3.5インチドライブのみ)や1.44Mバイトディスクへの対応が行われている。従来のIOCS.Xによる表示関係の高速化ルーチンも標準で搭載している。互換性という点ではDOSコール，IOCSコールを使ったプログラムなら当然，動くようになっている。

目立った拡張といえばファイルの高速化だろう。ファイル関係の外部コマンドとして，

FASTOPEN
FASTIO
FASTSEEK

といったコマンドが新設されており，オプション指定により，ファイル名21文字まで認識したり，大文字と小文字を区別して扱うこともできる。MS-DOS ver.4以降の同名コマンドと同等のものと思われる。

68882コプロセッサ

68020以降のCPUでは，数値演算コプロセッサなどが接続できるように拡張プロセッサ命令がサポートされた。X68030ではボード上に68881/68882用のソケットが用意されている。68881/68882のいずれかが使用可能だ。ただし，これまでX68000で使われていたPGAパッケージのものは使用できないので注意。

シャープ純正の周辺機器としては68882が提供される。X68000では数値演算プロセッサだったものが，X68030では数値演算コプロセッサとなる。これはCPUから直接制御可能になったことを表している。これまでのfloatN.xによる数値演算では数値演算プロセッサを高速なものにしても，CPUとのインタフェイス部分が遅いためまったく結果が変わらないという症状があった。これをコプロセッサとして接続することにより，計算自体の速度は同じでも値の受け渡しなどで全体に高速化されることになったのだ。昔，某所で「X68000ちゃんの200倍速い」といわれたMacintosh II がたかだか68020の16MHzに68881の組み合わせだったことを考えると，速すぎて困ってしまうが，そんなことはないので安心(？)してほしい。

今回のX68030では，浮動小数点演算ドライバとして従来のものとほぼ同様のfloat2.xとfloat4.xという32ビット専用ドライバが使用できる。拡張スロットに数値演算プロセッサを入れている場合にはfloat3.xも使用できるが……。

DōGAスタッフ７人に聞きました

X68030の新しいCGAデモを開発中のDōGAに，緊急取材。スタッフの意見を聞いてみました。

▶作画速度を比べると，XVIの３倍。初代X68000のコプロなしとX68030のコプロありとを比較すると，なんと16.75倍！　１時間かかっていた画面が３分半でできるということ。これは脅威的な進歩だ。初代機ユーザーで本格的にCGを始めたというユーザーには絶対買いのマシンだ。作画速度については68030用にチューンアップすることで，さらに高速になる可能性があるかも。(作品制作をガンガンやっているスタッフ)

▶速度と価格だけなら，IBM互換機なんかと比べて見劣りがしてしまう。だけどそれはあまり意味がない。なぜなら，自分のしたいことがX68000でしかできなければ，どんなにCPUが速いマシンを買ってもだめだから。(一時はPC互換機も検討していたスタッフ)

▶X68030はフルモデルチェンジというよりは，ただ高速化を図ったX68000のマイナーチェンジといった感じだ。CPUが変わったといってもマルチタスクができるわけじゃないし……。でも，中途半端なバージョンアップで互換性がなくなるよりはいい。X68030はX68000の究極バージョンだと思う。(ぜんぜん違うマシンが出ると期待していたスタッフ)

▶中身のボードを見ると……。今の段階ではいえないけど，いろいろと面白いことができそうだよ〜ん。(X68000を24MHzにしたスタッフ)

▶完全に互換性がとれないのは残念。過去のソフトのうち動かないものが結構出てきそう。DōGAではKo-WINDOWが動かなかった。でも，もう直しました。ついでに，68030対応のREND030.Xも完成！(RENDXVIの作者)

▶EPA2がサクサク動くよ〜。うれしいよ〜。CADの透視図も，かなり面数の多い物体でも“いつもより，多く回しております”てなもんだい。PCM8を常駐させても重くならないし。私はEXPERTのユーザーですが，このマシンは“買い”ですっていうか，もう“買います”。(EPA2の作者)

▶フニャー (暖かいマシンの上で寝ている猫)

機種	MPU	クロック	コプロセッサ	プログラム	実行時間	速度比
X68000 ACE	68000	10		REND.X	16m34s77	1.00
X68000 XVI	68000	16		REND.X	10m20s12	1.60
X68000 XVI	68000	16	68881	RENDXVI.X	3m08s32	5.28
X68000 XVI	68000	24	68881	RENDXVI.X	2m01s32	8.20
X68030	68030	25		REND.X	3m28s60	4.77
X68030	68030	25	68882	REND.X	2m50s91	5.82
X68030	68030	25	68882	REND030.X	59s38	16.75

作画ベンチマークテスト結果 (1992年11月号参照)

●充実のウィンドウ環境

SX-WINDOWはようやく実用段階に達した。大幅な機能アップと高速化を行ったSX-WINDOW ver.3.0と強力な32ビットマシンによって達成された新環境だ。

グラフィックが65536色に対応し，画像フォーマットとしてPIX以外にPIC，JPEG，TIFFなどがサポートされる。

CGAウィンドウというのが開き，動画も扱えるようになる，そうだ。最近のウィンドウシステムでは画面の中で小さな絵を動かすのが流行っているらしいので，それに迎合したものだろう。X68030なら秒間30コマでアニメーションを行うこともできるらしい。QuickTimeに対応するとかの話を聞いたので，扱えるデータが増えるのならそれはそれでよいことであろう。しかし，扱うデータの量や，メディアを考えると“？”な面も多い。

カラー化もいいが，なんといっても今回の目玉はマルチフォント日本語エディタ「シャーペン.X（仮称）」だ。PAT4形式のグラフィックを張り込め，禁則処理に対応し，マルチフォントで文書が書ける。洗練されてはいないが，細かい指定のできるインタフェイス。これならSX-WINDOWに移ろう，と思わせる逸品だ。使っていて楽しくなる類のプログラムである。

●……あのASK68Kが！

そして，これを支えるのがASK3.SYSだ。試しに打ち込んだ新聞の投書欄で，ほとんど誤変換のない性能に編集室は色めきたった。むしろ小さめな辞書，ファイルサイズで50Kバイト，常駐量で40Kバイトものシェイプアップがなされ，しかも単なる漢字変換プログラムを超えたアクセサリインタフェイスを備えている。たとえば文書作成中に変換中の語をキーとして，データベースを検索したり，電卓を呼び出したり，文法チェックやスペルチェックを行ったりするアクセサリが起動できるのである。これ自体がすでに環境を持っているのだ。

＊　＊　＊

まだまだ説明し足りない部分が多い。新しく加わった機能はほとんどが従来機種でも使用できるものとなっている。今回の新製品の焦点は強力なハードウェアと大幅強化されたソフトウェアの２面で構成されている。従来機種でも多大な恩恵を享受できるだろう。新システムは気合十分。

新マシンのパワーは押し入れの隅に眠っていたソフトを賦活することだろう。

もう逆走なんて呼ばせない。

表　X68030の仕様

	CZ-500C	CZ-510C	CZ-300C	CZ-310C
CPU	MC68EC030-25(25MHz) 80C51(キーボードスキャン／テレビコントロール用)			
ROM	IPL，BIOS　128Kバイト キャラクタ用ROM　768Kバイト 16×16ドット，24×24ドット　全角(JIS第1／第2水準漢字) 8×16ドット，12×24ドット　半角 8× 8ドット，12×12ドット　1/4角			
RAM	メインメモリ　4Mバイト　最大12Mバイトまで増設可 テキスト用VRAM　512Kバイト(ビットマップ方式) グラフィック用VRAM　512Kバイト(ビットマップ方式) スプライト用VRAM　32Kバイト スタティックRAM　16Kバイト			
表示能力 実画面エリアサイズ	テキスト　1024×1024ドット　4プレーン グラフィック　1024×1024ドット　4プレーン 512× 512ドット　16プレーン			
表示能力 表示画面モード テキスト表示	●実画面エリア　1024×1024ドット時 高解像度モード　768×512ドット 640×480ドット 512×512ドット 512×256ドット 256×256ドット 標準解像度モード(オーバースキャン)　512×240ドット 256×240ドット 512×480ドット(インタレース) 各モードとも ドットごとに65536色中任意の16色を指定可能			
表示能力 表示画面モード グラフィック表示	●実画面エリア　1024×1024ドット時 高解像度モード　768×512ドット 640×480ドット 512×512ドット 512×256ドット 256×256ドット 標準解像度モード(オーバースキャン)　512×240ドット 256×240ドット 512×480ドット(インタレース) 各モードとも ドットごとに65536色中任意の16色を指定可能			
表示能力 表示画面モード グラフィック表示	●実画面エリア　512×512ドット時 高解像度モード　512×512ドット 512×256ドット 256×256ドット 標準解像度モード(オーバースキャン)　512×240ドット 256×240ドット 512×480ドット(インタレース) 各モードとも 1) ドットごとに65536色中任意の色指定可能(1面) 2) ドットごとに65536色中任意の256色指定可能(2面) 3) ドットごとに65536色中任意の16色指定可能(4面)			
表示能力 スプライト	●パターン定義 サイズ　16×16ドット/パターン，8×8ドット/パターン 定義数　128パターン(バックグラウンド2面未使用時最大256パターン) 色　1パターンにつき16色/65536色(ドット単位) ●表示 スプライト座標系　1024×1024ドット 表示画面　512×512ドット(バックグラウンド1面表示) 256×256ドット(バックグラウンド2面表示) 表示制限　128スプライト/画面，32スプライト/ライン			
表示能力 特殊機能	スムーススクロール(テキストは円筒，グラフィックは球面)/特殊画面制御機能/ プライオリティ機能/パレット機能/半透明機能／スーパーインポーズ機能			
サウンド機能	FM音源　ステレオ8オクターブ　8重和音同時出力 音声合成　AD PCM(Adaptive Differential PCM)			
フロッピーディスクドライブ	5.25"FDD　2基搭載		3.5"FDD　2基搭載	
	オートロード/オートイジェクト			
ハードディスク	内蔵可能	80MB内蔵	内蔵可能	80MB内蔵
入力装置	ASCII準拠フルキーボード マウストラックボール同梱		ASCII準拠コンパクトキーボード マウス同梱	
インタフェイス	プリンタ(セントロニクス社仕様に準拠)/ジョイスティック(2個)/ テレビコントロール/アナログRGB出力/オーディオ入出力/RS-232C/ 外部フロッピーディスク/マウス/キーボード/イメージ入力／SCSI			
拡張I/Oスロット	2スロット内蔵			
専用ソケット	増設RAM専用ソケット			
電源/消費電力	AC 100V 50/60Hz			
	38W	40W	38W	40W
外形寸法(幅×高さ×奥行)重量	本体　155×363×270mm		本体　155×363×270mm	
	7.8kg	8.0kg	4.2kg	4.4kg
	キーボード　463×33×196mm 1.5kg		キーボード　380×38×170mm 0.95kg	
	マウストラックボール　73×32×105mm 0.14kg		マウス　63×37×97mm 0.11kg	
付属ソフト	オリジナルウィンドウシステム(SX-WINDOW Ver.3.0) オリジナルOS(Human68k Ver.3.0)/オリジナルBASIC(X-BASIC Ver.2.0) 日本語マルチフォントエディタ，グラフィックパターンエディタ 日本語フロントプロセッサ Ver.3.0			

アクセラレータを作る（第3回）

制御線の変更と信号のつなぎ方

Ishigami Tatsuya 石上 達也

今回も予定を変更して，回路図のちょっとした変更点の解説と補足説明を行います。さて，X68030が発表され，新しいHuman68kも登場しました。68000でも68030でも動く優れものです。ソフトはこれでかなり楽になるかな……。

いま，私は卒業論文というものを書いています。大学4年生になると，襲いかかってくるアレです。こっちから進んでいかなくても，向こうから勝手にやってくる，まるで台風のような奴です。

避けようがないので，このやっかいな災害をなんとか世間の役に立つ使い方ができないだろうかと日々思索を重ねています。

その成果の一例を紹介しましょう。

（家庭で）
「最近，帰りが遅いんじゃない？」
「うん，卒論だもの」
（友人と電話で）
「終電，逃しちゃった。今夜泊めてくれない？」
「えぇー，またかよ」
「卒論が忙しくてね」
（恋人と）
「電話かけても，家にいないじゃない」
「いま卒論書いているんだ」
（編集部で）
「連載の進度，遅いよ」
「すみません。卒論なもので」
というような具合です。

生涯でたった一度だけ使用が許された免罪符。それが，卒論です（注：文系の学生が使ってはいけません）。

と，なにかのいいわけにも使える反面，使い方を間違えるとなかなか厄介です，これが。シャレじゃなくて本当に睡眠時間が減ったり，帰宅が遅くなったり，終電を逃がす日が続きます。あと，昼と夜に外食するから出費のほうも馬鹿になりません。

と，そういう状況なので，今月の実験成果はまったくありません。「あまり」でも「ほとんど」でもありません。「まったく」です（あはは）。

回路製作のほうが進んでいないので，そっち方面の話はないのですが，いくつか話していなかったことが残っていたので，そちらのほうを消化してしまいます。

先月からの変更箇所

アクセラレータ上に68HC000を残しておいたのには2つの理由がありました。ひとつはEクロック発生装置としての役割。もうひとつは，将来，スイッチひとつで68000モードと68020モードの切り替えをできるようにするための布石。

そして，その切り替えをバスマスタ制御信号を操作することによって行おうと当初は考えていました。バスマスタ制御信号というのは，DMAコントローラのようにMPUのかわりにシステムを乗っ取ってしまうような装置のためのもので，この制御信号を出されるとMPUの動作に「おあずけ」がかかります。

この「おあずけ」は，動作を一時中断させるだけでなく，MPUの持っている権利の大部分を放棄させる働きがあります（厳密にいうと，ほとんどの出力信号がハイインピーダンス状態になる，という。ハイインピーダンス状態とは抵抗値が無限大になることで，つまりは電気的に切り放されること）。この「おあずけ」をかけることによって，DMAコントローラ，MPUに無断でメモリ内容などを書き換えることができるようになります。

具体的には，これらの制御信号は$\overline{\text{BR}}$（Bus Request：バスマスタ権を取りたいデバイスが，MPUにその旨を主張する），$\overline{\text{BG}}$（Bus Grant：MPUがバスマスタ権を放棄したことを表明して出力する），$\overline{\text{BG ACK}}$（BusGrant Acknowledge：その結果，バスマスタ権を収得したデバイスがこれを出力する）の3本から成ります。この方法を3-Wired Arbitorationといいます。

もうひとつMPUを一時的に切り放す方法に$\overline{\text{HALT}}$信号をアサートする，という方法がありました。バスマスタ権をMPUから取り上げるというのが「おあずけ」的な使い方だったのに対し，この$\overline{\text{HALT}}$信号をアサートするというのは「休め」的な意味あいです。具体的には，ハードウェアの具合が悪くなったとか，仕事がひととおり終わったのであとは電源を切られるのを待つだけ，というような場合に使います。

アクセラレータ回路の制御方式に，なぜ$\overline{\text{HALT}}$信号を使わなかったかというと，これを使うと68HC000のEクロック出力がキャンセルされるような気がしたからです。マニュアルには，一応「Eクロックがハイインピーダンスになるか→No」みたいなことが書いてあるのですが，テストしてないので実際のところはわかりません。

先月号の実験結果を見てもわかるように，X68000の内部ではEクロックを一切使用していないので，そんなことはどうでもよいことだったのです。

そんなわけで，3本の制御信号をハタハタさせるよりは1本の制御信号を操作するだけの方法に変えてみました（図1）。

自分でいうのもなんですが，この連載は終了するまでハンダごてを握らないほうが正解みたいです。

ちょっとアナログ的なこと

TTLで回路を組むときの鉄則に，電気は一方向に流す，というものがあります。道路標識でいうと，すべての道が一方通行なのです。一方通行ということは，入り口があり出口があります。

入り口同士を結んでも問題はありません。中学校で習ったとおり導線で結ばれているところは電圧が等しいですから。

さて，出力同士を結ぶと，どのようなことが起こるでしょうか？　2か所以上から信号が出力されてくるわけですから，信号がかちあったり，下手をすると信号の逆流（つまり，出力端子からほかのTTLの出力信号が入り込んでくる）が起きてしまいま

す。そして世の常として，こういうことをやるとヤワな半導体製品というのは「飛んで」しまいます。

普段はこういうことをやりたくないのですが，どうしても出力同士をぶつけなければならないときというものがあります。よくある例として，「どこか1カ所でも信号がLOWになったら」という論理和回路があります（図2）。この回路は，どこにも論理和演算用の半導体がない代わりに配線によって論理和演算回路を構成しているという意味で，ワイヤードORと呼ばれています。X68000ではメモリアクセスに対し割り当てられたデバイスが応答反応を返す部分（信号名 $\overline{\mathrm{DTACK}}$：Data Transfer ACKnowledge）で使われています。このように，どうしても，というようなときに使われるのがオープンコレクタタイプのTTLです。

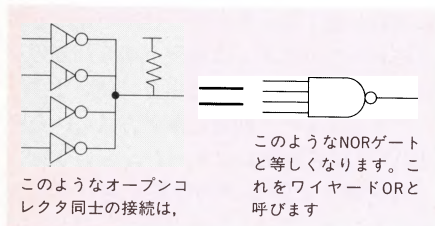



図2　ワイヤードOR

このようなオープンコレクタ同士の接続は，

このようなNORゲートと等しくなります。これをワイヤードORと呼びます

今回使用したのは74LS06というオープンコレクタタイプのNOTゲートです。普通，NOTゲートに使うのは74LS04ですので，これらの比較をしてみましょう。

データシートを見ると「7405の高耐圧出力型（30V耐圧）」と書かれていて，7405のほうを見ると，「7404のオープンコレクタ型」と書かれています。なぜ，高耐圧出力型かというのはひとまずおいておき，このオープンコレクタ型ということで7404と7405の違いを調べてみます。

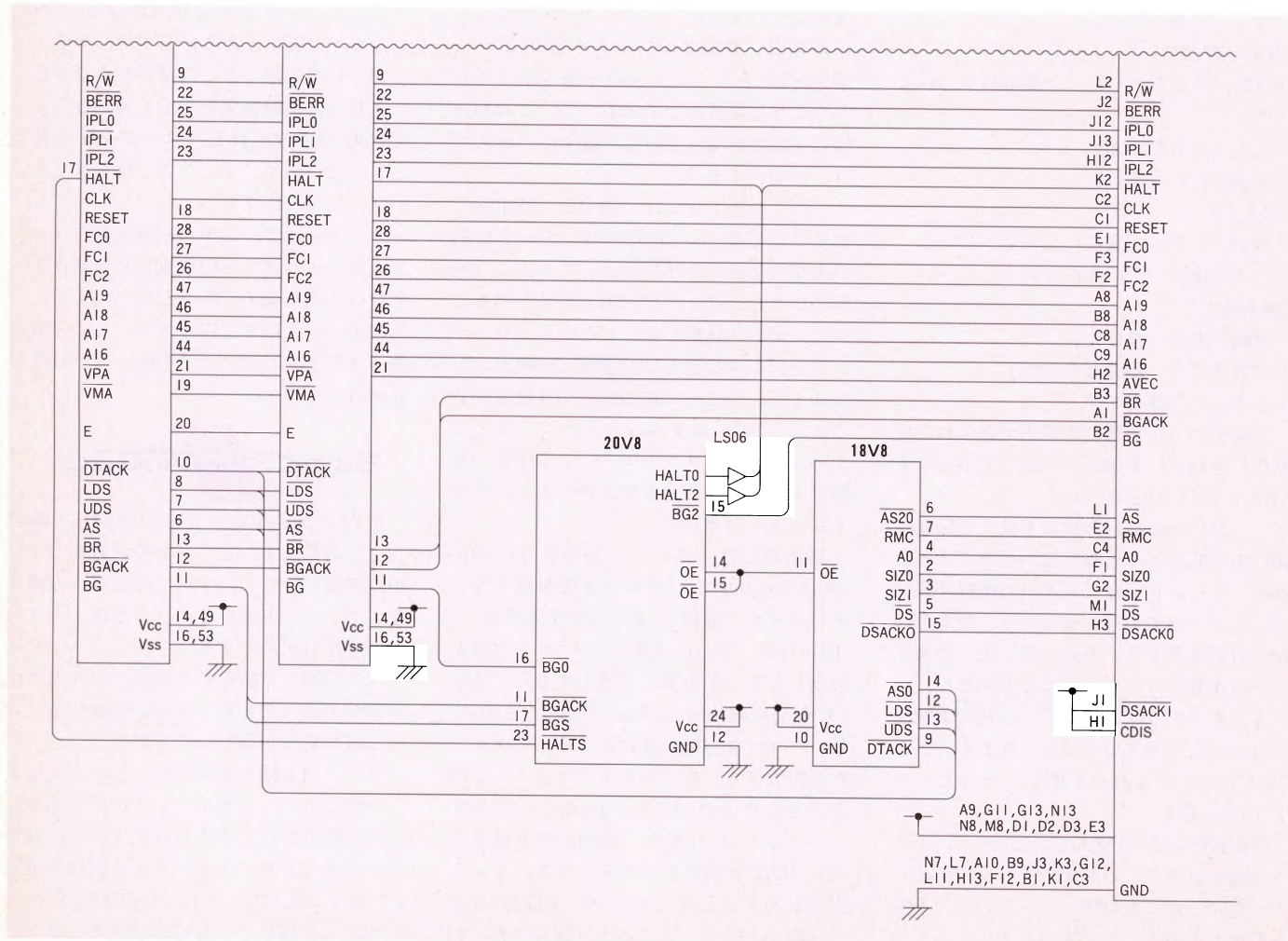
図3がこれらの2つのTTLの内部等価回路です。74LS05のほうはトランジスタが2つばかりのシンプルな構成で，最終段階のトランジスタのコレクタから直接出力端子が伸びています。コレクタ端子が開放されている，というのでオープンコレクタ端子。実に明快な命名です。

これに対し，74LS04のほうには，なに使っているのかよくわからないトランジスタやらダイオードが見受けられます。

つまり，このよくわからない部分が電気の逆流によって「飛んで」しまうのです。

じゃあ，この飛んでしまう部分はいらないものなのか，ということになります。そうではありません。

オープンコレクタタイプの信号はLOWになるのは早いですが（なんたってベースに電流を流すだけですから），HIGHにするには時間がかかります。図3でいうと，この回路に出力がHIGHということは，すべてのTTLの出力はハイインピーダンスになったというわけで，頑張っているのはプルアップ抵抗ひとつだけです。つまり，この回路で出力信号がLOWからHIGHになる瞬間というのは，すべてのTTLが「おれは関係ないぜー」と電気的に関係を切り放したうえで，ひとり残されたプルアップ抵抗が「しょうがねーなー」と＋5Vから出力端子に電流をシコシコ送っている状態なのです。そして出力端子に電流が送られると，そこの電位は徐々に上がっていき

図1　「今月の」新しい回路図

ますから，やがてスレッショルド電位（しきい値電圧）になってHIGHになるのです。

つまり，オープンコレクタタイプのTTLというのは出力端子同士を結べるけれども，LOW→HIGHの変化が遅いという性質を持ったものなのです。

と，ここまで書いて気がついたのですが，以上の話は，7400と7403（NAND），7404と7405（NOT），7408と7409（AND）には当てはまりますが，今回のアクセラレータに用いた74LS06には当てはまりません。7405の改良型ということで，きっとメーカーの人が頑張ってくれたのでしょう（データブックによると，H→L：15nsで，L→H：20nsだそうです。パチパチ）。

オープンコレクタタイプで実際に問題になるところ（HALTの場合）

HALTを操作することにしたのに，図1を見ると直接GALからHALT端子へ信号線を伸ばしていません。あいだに74LS06（先ほどのオープンコレオクタタイプのNOTゲート）が噛まされています。これは電流の逆流からGALを守るためのものだったのです。

またまた余談になりますが，68000のHALT端子に，なぜオープンコレクタタイプのTTLをつなげなければならないのでしょうか。だって，そうでしょ。普通に考えれば，HALT端子ってのはMPUをHALTさせるための入力信号なんだもの。電気の逆流なんて，ありっこないじゃない。

で，その解答が図4です。なんと，68000のHALT端子は，直接LEDなどを駆動できるように，そのための電力をも供給してくれるのです。信号だけではないのです。電力なのです。電力。

確かに，こうすれば部品数をいくらかは削減できそうですが，このようなことは，さすがに最近は流行らないらしく，68000ファミリも68030からはやっていません。今回の製作では，そんな端子にGALなどというヤワなICをつなぐわけにはいかないので74LS06を挟んであるのです。

オープンコレクタタイプで実際に問題になるところ（DTACKの場合）

先ほどもちょっと話しましたが，68000にはDTACKという入力端子があります。この信号はメモリアクセスに対する応答信号で，これが入力されない限りMPUはデータバスの値を読み込んだり次のサイクルへいったりというようなことを行いません。

X68000の場合，いくら待ってもこの信号が入力されなかったということは，そのアドレス空間にはメモリなりデバイスなりが接続されていないということですから，この場合はBERRを代わりにシステムが発生し，「バスエラーが発生しました」ということになります。

X68000はDTACKという端子がオープンコレクタタイプのバッファで入力すべし，となっています（実際にカスタムLSIなどを調べたわけではありません。増設メモリボードを作っていて，この信号を普通のバッファを通して返したら，ほかのボードにアクセスできなくなってしまったのです）。

74LS06などは別としても，一般にオープンコレクタタイプの信号はL→Hの変化が遅いのです。これをLow Active（Lowレベルで有効）の信号に使うと信号のキレが甘い，ということになります。一度，信号を有効にしてしまうと，それがダラダラと続き，無効にするまでの時間がやたらとかかってしまうのです。

それでも，10MHzぐらいなら問題にもならないのですが，せっかく68020を新しいMPUに使うのですから，動作クロックはMPU部分だけでも20MHz以上を使いたいものです。そして，このとき問題が起こるのです。

以下ではアクセラレータボードがうまく動作し，MPU部分だけの高速化といった，あとに続く実験を行えるようになる，という仮定の下に話をします。

MPUがメモリアクセスを行い，メモリがデータとともにDTACKを返します。ここまでは問題はありません。これでメモリアクセスサイクルの終了です。そしてMPUが内部処理を行い，次のメモリアクセスが始まります。MPUがアドレス信号を出力し，それにメモリが応答してくれるのを待ちます。少なくとも待とうとします。

しかし，DTACKというのはオープンコレクタタイプのバッファを介してMPUに届いていますから，LOW→HIGHの変化に時間がかかるのです。つまり，前のメモリアクセスサイクルで，一度有効になったまま，無効にならないうちにMPUに読み込まれてしまうのです。その結果，MPUは本当のメモリアクセスを待つことなく，メモリアクセスを始めた途端に，応答信号が返ってきて，まだ過渡状態にあるデータバスの内容が取り込まれてしまうのです。

MPUはまったくデタラメなデータを読み込むことになりますから，暴走くらいしかすることがなくなってしまうのです。

弱った，弱った，というわけで解決方法です。問題は次のメモリアクセスサイクルが始まっても前のサイクルの応答信号が残っている，というところにあるわけです。つまり強制的にメモリアクセスサイクルの初めに応答信号を切ってやればよいのです。

運がよいことに68020ではメモリサイクルの初めに/ASが入り直します。一瞬ネゲートされてから，アドレスバスが確定して再びアサートされます。これを利用して，メモリサイクルの始まりを検出してDTACKのネゲート遅延を解消してやります。その回路が図5です（参考文献2より引用）。

＊　　＊　　＊

今後の予定ですが，来月号はお休みをいただいて，再来月号あたりには復帰します。

参考文献

1）最新TTL IC規格表，CQ出版

2）細田誠，68000系ハードウェア設計ガイド，CQ出版

図3　74LS04と74LS05の内部等価回路

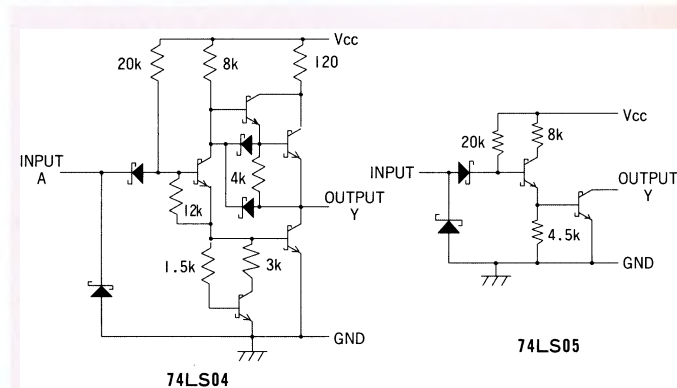



図4　HALT端子でLEDを駆動する

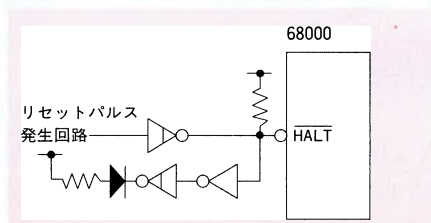
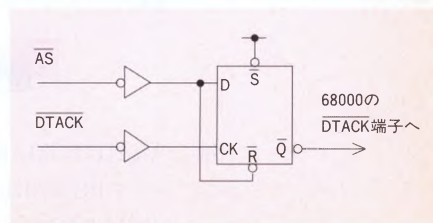



図5　DTACKのネゲートの遅延を打ち消す回路

CGAマガジンの積極的な使い方(その2)

プロジェクトチームDōGA
かまた ゆたか

先月に引き続いて，CGAマガジンのデータを利用して，F1がシケインを走り抜けるシーンを，1カット制作してみます。これを修得すれば，オリジナルCGAもガンガン作れるはず！

はじめに

大晦日に一首，

　マウス持ち　耳傾ければ　除夜の鐘

　CGAマガジン　いつできるやら

3月号で正月の話もなんですが，今年の正月はなかなか悲惨でした。発行予定日はとっくに過ぎているのに，いつまでたってもCGAマガジンは完成しない。いったいどうなってるんだと聞いてみると，編集長は敵前逃亡している。尻拭いモードに入ってみると，データは未完成，編集は全然，システムはバグだらけ……。急遽，編集部員を増員し，年末年始を無視した突貫作業に突入！

CGAマガジン創刊号の発行が1カ月遅れたこと，ならびに2枚組となったため，TAKERUでの配布価格が600円値上げされたことをお詫び申し上げます。

でも，仕上がりは結構よかったんではないかと思っているのですが，皆さん楽しんでいただけましたか。「お試しシステム」で作れるCGAよりはかなり本格的でしょう。ご意見，ご感想など，ぜひお便りください。

さて，CGAマガジンで，ほかの人が作ったCGAを実際に見るだけでなく，そのデータを活用して，オリジナルのCGAを作ってみようというのが，CGAマガジンの積極的な使い方です。2月号では，CGAデータベースから，道路とF1のデータを展開し，1，2カットのモーションデザインをしてみました(ちゃんとできましたか？)。今回は，このカットを仕上げてみましょう。

前回は，「その1」ということで，初心者向きに詳しく解説しましたが，今回は実践編ですので，各ツールの基本的な操作は理解しているものとして，あまり細かい操作は省略します。わからない点などありましたら，これまでの連載やマニュアルをよく見直してください。

遠景を作る

遠くの背景はBGMAKE.Xで作ります。今回のバージョンアップでBGMAKEもかなり使えるものになりました。このBGMAKEは，ペイントソフトなどで描かれた12枚の画像を背景とし，視線ベクトルに応じてその一部を切り出します。ですから，まずこの12枚の背景画像データを用意することから始めましょう。

○背景データを展開する

自分でこの12枚の画像を描いてもよいのですが，面倒なのでCGAマガジンに収められている背景用画像データを利用します。「CGAデータベース」の「データベース一覧」の後ろから3つ目「背景データ関連・F1用背景」を展開してください。「SKYa001.PIC」～「SKYa012.PIC」の12枚の画像ファイルが用意されます。

展開のときによく注意して見ているとわかるのですが，実はこのデータは3枚分の画像しかありません。「SKYa001.PIC」と「SKYa005.PIC」と「SKYa009.PIC」です。残りの「002」～「004」は「SKYa001.PIC」を，「006」～「008」は「SKYa005.PIC」を，「010」～「012」は「SKYa009.PIC」をコピーして使っています。つまり，東西南北4方向の背景はすべて同じ画像ということです。手抜きも甚だしいのですが，実用面で問題があることは少ないでしょう。問題がある場合は，この絵を下地にして各自で描き加えてください。

○BGMAKEの実行

2月号で制作した「X02A.FSC」と「X02B.FSC」の視線に合わせた背景を作るので，まず「X02A.FSC」と「X02B.FSC」を用意してください。BGMAKEは入力ファイルとしてフレームファイルが必要です。ですから，BGMAKEを実行する前に，この2つのフレームソースをFFにかけます。そこで，

```
FF X02A
FF X02B
```

とします。

BGMAKEの書式は，

```
BGMAKE<フレームファイル><背景画像名>/o<出力画像名>
```

となっています。<背景画像名>は，画像ナンバーを除いた部分を入力します。つまり今回は「SKYA」です。

```
BGMAKE X02A.FRM SKYA /oBACKA
BGMAKE X02B.FRM SKYA /oBACKB
```

とすると，しばらく，

Making background picture ...
と画面表示され,「X02A」と「X02B」用の背景動画(「BACKA001.PIC」~「025.PIC」,「BACKB001.PIC」~「020.PIC」)が出来上がります。

できた画像を見ればわかりますが,ほとんど空は見えず,地面ばかりの絵になっています。これは,視線が見下ろすような角度になっているからです。また,山々の絵の解像度がかなり粗くなっていますが,これは画角が15度とかなり狭くなっているため,背景画のほんの一部を256×256に引き伸ばすことになっているためです。遠くなので,ピントがずれていることにしましょう。

○F1の画像と合成する

ここで先月作画した画像と合成してみましょう。その前に,だんだん画像データが多くなってきたので,ハードディスクの中身を整理しておきます。内容ごとにディレクトリを作って,画像データを移します。

```
MD BACK
MOVE SKYA*.PIC BACK
MOVE BACK*.PIC BACK
MD X02
MOVE X02*.PIC X02
MD TEST
```

このへんの操作はCGAシステムのマニュアル「CGA大学¥コンピュータ基礎概論」などで勉強して,ちゃんと理解しておいてください。やはり,本格的にCGA制作を目指すなら,パソコンの基礎も身につけておくべきです。

さて,画像を合成するのには2通りの方法があります。まずは,きわめて短時間にできるTPILE.Xを使ってみましょう。

```
TPILE BACK¥BACKA X02¥X02A /OTEST¥TES1A
TPILE BACK¥BACKB X02¥X02B /OTEST¥TES1B
```

この方法は単純に2種類の画像を重ねて出力しているだけです。ですから,F1や道路の縁に黒い点々が残ってしまいます。これはアンチエイリアス上の問題です。

この黒い点々がどうしても気に入らないのでしたら,背景と合成しながらもう一度作画し直すしかありません。

CGAコンテスト事務局より

いよいよ第5回アマチュアCGAコンテストが間近に迫ってきました。作品もすべて出揃って,すでに審査に入っています。そこで,作品をいくつか紹介しながら,今回の見どころなどを解説しましょう。

今回の最大の見どころは“新人の台頭!”です。昨年,CGAシステムを再配布したことに関係あるのかどうかわかりませんが,今回は応募総数が倍増しています。そして,これからのアマチュアCGAをリードしていくだろう新人たちがたくさん入選しました。私は,今回がCGAコンテストのひとつの転換期になるのではないかとにらんでいます。

まずは「MISSION」。大阪工業大学グラフィックス研究会(GR)の浅井さんの作品です。この作品はいわゆるバトルロボットものです。しかし,従来のバトルロボットとは一線を画しています。いままでのバトルロボットは,単にロボットを作って動かしてみましたという感じだったのですが,「MISSION」では,本当に熱いバトルを展開します。いったい,このアクションはどのようにしてつけたんでしょうか。

なんでも,いままでコンテストに応募していなかったのは,このアクション(特に下半身)の動きを表現する技術を開発していたからなんだそうで,今後もそういったアクション物の作品を発表するとおっしゃっていました。いったいどんな技術(プログラム?)なんでしょう。コンテストが終わったら公開していただけないかな。ということで,「MISSION」のアクションは要チェックです!

「面会」「ハッピー バレンタイン」の2作を応募くださった客野さんは,イラストレータ兼業主婦ということで,第2の寺尾響子さんという感があります。作品はキャラクターデザインや色づかいが,さすがプロという仕上がりになっています。特に「面会」では,作品の完成度が非常に高くなっています。

今回はX68000以外の機種も頑張っています。山畑さんの「Answer The Door」は「RAY-TREK II」という業務向け使用にも耐えうる高級レイトレーシングソフトを使用しているだけあって,画質のクオリティは高いですね(どうやってビデオに落としたんだろう?)。この作品のいいところは,単純明快に短くまとめられていることです。木が数本生えているだけのだだっ広い空間に,ドアがひとつ。いったい,何が出てくるんだろうと近づいてみると……。

布山さんの「Complex」はMacintoshの「Macro Mind Director」で制作されています。白と黒の2色,2本の直線があるだけのシュールな画像。音楽もありません。しかし,この2分間の作品はとっても楽しんで見ることができます。このあたりは素晴らしいセンスですね。間の取り方などが非常によいのです。2本の直線は片方が短いことを気にしているようで,なんとか同じ長さになろうと努力します。

新人が伸びているなか,大御所の方々はどうなっているんでしょう。入賞の常連,芸術賞独占の宗戸さん,EPA2の宇宙人森山さん,オタッキーCGAを目指す西之園さんらは今回出品しておらず,なんとも寂しいところです(もっとも,宇宙人森山さんは実はオープニングアニメーションを制作してくださっているのです)。そんななか,健在,いや以前よりさらにパワーをつけているのがKMC上原さんの「マウス」,もうひとりの森山さんの「SWORD2」です。

2年前,第3回CGAコンテストでこの2人がグランプリを競ったことは,まだ記憶に新しいと思います。そのときは僅差で森山さんの「SWORD」がグランプリを受賞しました。今回の「SWORD2」は「SWORD」のリメイクバージョンですし,「マウス」は「CLOCK」と同じ路線ですので,まさにあの2年前の戦いが再現されるわけです。森山さんが2度目の受賞となるか,あるいはKMCが悲願の初受賞となるか? ちょっと注目したいと思います。

大御所のグランプリ候補としては,昨年のグランプリを受賞した「猿蟹合戦」の宍戸さんの「A PLANET」,一昨年に映像賞を受賞した「おはようございますの帽子屋さん」の小島さんの「ある夜の出来事」なども見応えがあります。さあ,いったいどうなるんでしょう。

さらに,今回から設けられた「1カット部門」「4カット部門」も見逃せません。この部門は,一般部門に応募するにはいたらない初心者の発表の場を設けるという意図で設けられたはずですが,蓋を開けてみると“この作品のどこが初心者やねん!”というものも多く,一部の噂ではこっちの部門からグランプリが出るとまでいわれています。特に,「OBJECT:MECHANICAL HOUND」「DRIVIN' WOMAN」「不思議な煙突」などにはびっくりすること間違いなし! ビデオを入手した方は,思わず巻き戻して見ることでしょう。それから,コンテスト事務局内で制作した冗談作品「サンドストーム」も笑えます。みんな,作品としては短いので,ここで種明かしするわけにはいきませんが,お楽しみに。

上映会の場所,日時ですが,今年は場所が違うから注意するんだぞ~。今年は新宿だ!

日時:1993年3月14日(日)
　　開場 PM1:30 開演 PM2:00
場所:新宿朝日生命ホール(JR新宿駅西口前
　　徒歩2分 朝日生命ビル2階)

この場合，当然作画時間がかかります。

REND /A2 /G /HBACK¥BACKA001 / OTEST¥TES1A STRAI.SUF R30.SUF R10.SUF ROAD.ATR WILLI.* BENET.* X02A.FSC

REND /A2 /G /HBACK¥BACKB001 / OTEST¥TES1B STRAI.SUF R30.SUF R10.SUF ROAD.ATR WILLI.* BENET.* X02B.FSC

「/H」は合成する画像を指定するオプションです。でも，どうせテストなのですから，ここまでする必要はないかもしれません。

アニメーションさせてみる場合は，

MKTCH TEST¥TES1A001 TEST¥TES1B002

を実行して，タイムチャートファイル「TES1A.TCH」を作ります。

256色を超えているので本当はCRDを使わないといけないのですが，テストに時間をかけるのも面倒なので，65536色モードでアニメーションさせてみます。

HANIM /M2 TES1A

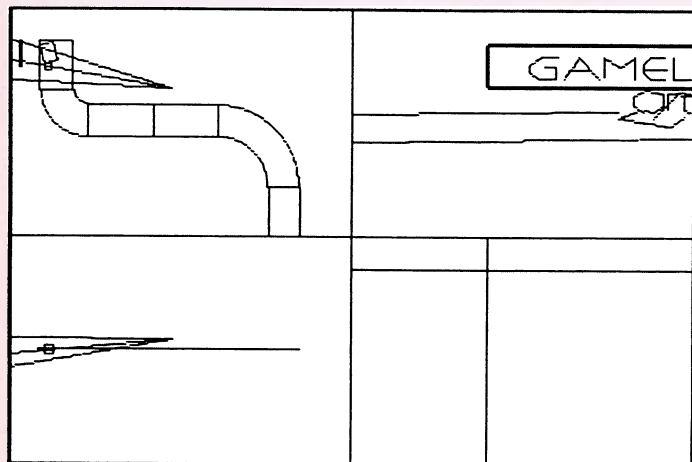
65536色モードでは再生速度が若干遅くなりますが，この程度の画像ならX68000 XVIではちゃんと毎秒20フレームで再生できます。

作画を行わなかった方は，「Oh!X Graphic Gallery」をご覧ください。この背景をつけるだけで，だいぶ雰囲気は変わりましたよね。

近景を作る

BGMAKEでつけた背景だけでは，まだまだ寂しいですね。特に手前の緑色の地面がべた塗りのままになっています。そこで，この地面を隠すように，いろんな物体を置いていきます。

○看板を配置する

図1　看板を置く

いちばん有効な物体が看板です。まずはこの看板を配置してみましょう。「CGAデータベース」の「看板各種」を展開してください。「SIGN1.SUF」～「SIGN6.SUF」の6種類の看板と共通のアトリビュート「SIGN.ATR」ができます。

看板を配置するのは，もちろんFFEで行います。

FFE X02A

として，前回のフレームソースを読み込んで起動させます。FFEの操作方法は，さすがにもう解説しません。いままでの連載で，十分使いこなせるようになっているはずです。マウスカーソルのカウント量を「100」に設定したり，「＝」で画面表示範囲を広くしたりすることも忘れないでください。

看板は6種類用意していますので，好みのものを使ってください。同じものを並べるより2種類交互に並べるほうが派手でよいでしょう（だからといって，3種類以上並べるのは不自然です）。その場合，色合いなどに注意してください。今回はウイリアムズの車体の色も考えて，「SIGN2」と「SIGN6」を使いたいと思います。

「物体設定/追加/SIGN2.SUF」を選択し，(－6000，2500,0)の位置で，Z軸に90度回転させてください。

そして，「作画」で確認し，「決定」します（図1）。以下同様に，

SIGN6.SUFを(－6000，1500,0)でZ軸回転90度
SIGN2.SUFを(－6000，　500,0)でZ軸回転90度
SIGN6.SUFを(－6000，－500,0)でZ軸回転90度

と配置していきます。できたら適当な名前でセーブして，終了してください。

○芝生を植える

地面の緑色の部分は草（芝生）が生えているつもりなのですが，実際の芝生は，緑の濃いところ，枯れかけているところなどがあって，不規則な模様になっています。これを表現するのが「SIBA.SUF」です。とはいっても，この「SIBA.SUF」は決してリアリティを追求したものではありません。ただ，1色でべた塗りする場合だと，カメラが動いてもその部分に変化がないのに対して，「SIBA.SUF」を使うと模様が動くので，心理的なスピード感がかなり違ってくるのです。

「SIBA.SUF」「SIBA.ATR」は「CGAデータベース」から展開してください。この「SIBA.SUF」は1辺が20m（±1000cga）の正方形となっていますので，道路の周りを20m間隔で適当に敷き詰めていけばよいのですが，少し問題があります。それは「SIBA.SUF」も「道路各種」もZ座標が0のため，重なると同一空間に複数の面が存在することになって，変な画像になってしまうという問題です。道路が直線の部分は重ならないように並べるとしても，曲がっている部分はどうしましょう？

この場合は「SIBA.SUF」のZ座標をちょっとだけ（5ぐらい）下げてください。そうすると理論的には，そこ

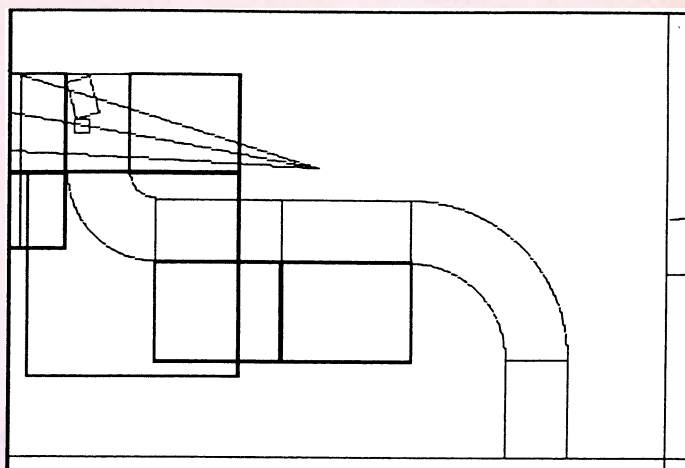
の地面だけがへこんでいるということになりますが，そんなことはちょっと見てもわかりません。

「SIBA.SUF」を並べる範囲ですが，道路周辺すべてを敷き詰める必要はまったくありません。画面から見える範囲だけで十分です。先ほどのアニメーションをよく見て，だいたいどの範囲が視界に入っているかをよく確かめてから，FFEを起動し，作業に入りましょう。

私は以下のように配置してみました（図2）。

位置：(−5500,　2000,　0)
位置：(−5500,　　0,　0)
位置：(−2000,−1500,　0)
位置：(　　0,−1500,　0)
位置：(−2500,　2000,　0)
位置：(−3500,−1000,−5)
拡大：X方向2倍，Y方向2倍

この6個目の「SIBA.SUF」が道路と重なっている部分です。見えている範囲すべてを覆っているわけではな

図2　芝生を植える

特別対談 どうやってあんなデータ作るの？

CGAマガジン創刊記念対談ということで，「TOMCAT」のデータを提供してくださった「チームTOSAKA」の野中さん，田中さんにわざわざ（？）九州から来ていただきました。

かまた：どうも，このたびはありがとうございました。“CADを使わせたら日本一”といわれるだけあって，「TOMCAT」は素晴らしい出来ですね。ノギスで計ったんですか？（笑）

田中：いえ，今回はサンプルですので，適当に“物差し”で大まかな寸法だけとりました。あとは適当ですね。やはりいちばん工夫したのは面数を減らしたこと，これに尽きます。

かまた：あれで面数が少ないんですか。はあ。200Kバイト近くあるんですけどねえ。作る前に，設計図やプラモデルを用意したのですか？

田中：ええ，いちおうプラモデルは用意しました。だいたい，プラモデルは部品ごとにバラバラになってますよね，だから，とてもモデリングしやすいんです。部品ごとにモデリングして，CADでくっつけると出来上がりますから。ただ，今回の一般民間人用サンプルデータは，なるべく面数を減らさないといけない。かといって，自分として表現したい最低ラインは作りたかったんです。ということで，この「TOMCAT」は，その妥協点の産物だと自分では思っています。

かまた：なにかTOSAKA独自のテクニックなどはあるのですか？

田中：そうですね，特に独自というわけではないのですが，今回は面数を減らさなければならなかったので，多面体を多数使用しています。面がつながらない場合もあるのですが，そのときはエディタで読み込んで，無理やり数値を書き換えています。±1ぐらいなら誤差の範囲みたいでつながっても問題ないようです。しかし，それ以上になると，シェーディングをかけた時点で変になりました。まあ，本当は，きちんとCADでつないだほうがいいとは思うんですが。それと，最初はあまり細部にこだわらずに，全体のスタイルを整えることから進めていったほうがいいと思います。

かまた：えっ，エディタで数値を書き換えるんですか。あの数値の羅列を見て，どこの部分かがわかるのですか？

田中：いや，当然わかりづらいですよ。だから，1面定義したら，データの読み込み，訂正，それの繰り返しです。人間，努力と根性です。

かまた：まあ，たしかに根性さえあれば，なんでもできますけど。

野中：根性のほかに，勇気もいります。

かまた：なんの勇気です？

野中：著作権に対する勇気です（笑）。

かまた：いったい，野中さんは現在どのようなものを制作しているのですか？

野中：ソボイアS.21試作戦闘飛行艇です。次回のCGAマガジン用にでもと考えています。

かまた：それって，“○の豚”の飛行機じゃないんですか？　著作権が心配ですね。本体を青く塗ってごまかしましょうか。

野中：以前，飛行機には著作権がないと聞いたので作ったのですが。

かまた：それは軍事用の戦闘機だって。

野中：これって，軍用の飛行機ではなかったんですか？

かまた：そういう意味じゃないって。ところで，この横のX68000のCADでは何を表示しているんですか？

野中：文月さんから依頼された「ステイメン」です。

かまた：私には「ガンダム」のように見えるんですけど。これはごまかしようがありませんね。どうしましょ。目の前に黒い長方形でも置きましょうか。

野中：それは一昨年の零式のときにやりました。それでもNG出したのは，かまたさんじゃないですか。

かまた：そうだったっけ。いやあ。

野中：そうそう，以前「ウルトラホーク1号」で，“色を変えたらいい！”とかいって，色を変えただけで載せたことがありましたよね，今回もそれでいきましょう。たとえば，目の色を変えるとか……。

かまた：にせウルトラマンじゃないんだから。でも，よくこんなに細かく作りましたね。

野中：ええ，もうマウスは使えません。キーボードで1ドットずつ動かして作ります。ポイントはパーツに分けて制作することですね。

かまた：でも，このガンダムの頭部なんて，パーツに分けようがないでしょう？

野中：いえ，前後2パーツに分けています。

かまた：そうすると，その2つのパーツのつなぎめがきちんとつながらないのでは？

野中：ちゃんと座標を紙に書いておきますから簡単です。

田中：私は，まず最初に断面図を作ります。この断面図のところでつなげば簡単です。

かまた：簡単……ですか。で，「TOMCAT」ではどのくらいの時間がかかっているんですか。

田中：2週間ぐらいですかね。ただ，平日は仕事がありますからあまりできませんが。

かまた：やはり結構かかるんですね。私が今回作ったF1なんか半日ですよ。手抜きですね。

田中：私はパーツごとに作って，組み合わせて一度AUTO.Xで作画してみて，各パーツごとに直していくという作業を繰り返します。実際には，作っている時間より描かせている時間のほうが長いです。

野中：私は，ひととおり最後まで作ってから，修正を繰り返します。

かまた：私は修正なんかしません。すいません。ということで，これからもすごいデータを作ってください。よろしくお願いします。

く，あちこち隙間もありますが，気にするほどのものではないでしょう。

○その他

これだけでも十分ですが，あとは好みに応じて並べてください。18フレーム目のあたりで，画面上部が若干寂しいので，何かを置くのもよいでしょう（タイヤバリアぐらい作っとけばよかった)。ただ，むやみやたらと置くと，メモリによっては作画できなかったり，メインのF1が目立たなくなって，うっとうしいだけになるので注意してください。

また，“道路の横何m以内に看板があるわけない”とか“F1のコースにこんな形のシケインはない”とかいった話も無視してください。事実をもとに作ったからといって，リアリティが出るとはかぎりません。むしろ，それで絵として間が抜けてしまったら，それだけの評価しかされません。CGAは映像であって，シミュレーションではないのです。

私はガードレール「GUARD1.SUF」をひとつつけ加えることにしました。

GUARD1.SUFの位置：(−5000,−1500,0)
Z軸回転　　　　　：−45度

ガードレールには表裏がある点に注意してください。支柱があるほう（Y軸の負）が道路の外側になるようにします。とはいっても，見てわからないようなら，どうでもよいことです。

ついでに，もうひとつ「EDGE.SUF」も置いてみまし

読者によるほっとけないほっとこらむ

受験シーズンも終盤に入りました。もうすでに合格が決まっている方もいらっしゃるでしょうが，まだの方はラストスパートに頑張ってください。

<Tさん(愛知県)>マニュアルの入金が遅れてすいません。払込用紙が行方不明になっていて，つい先ほど「チャート式基礎からの数学I 改訂版」の242ページから救出されました。今年は，受験の年。大阪大学に行って，肉体カンパをしてあげたいのですが，かねてからの野望を果たすために，北大理II（獣医学部）を目指しております。友人が大阪大学基礎工学部を受けるといっているので，そいつらを引き込んでこき使ってやってください。ちなみに高校は岡崎高校です。

うさ子：北大理IIといえば，「動物のお医者さん」で有名ですね。私も“チョビ”のスリッパを愛用しています。菱沼さんや漆原教授のモデルとなった方は本当にいらっしゃるのでしょうか？　私はなんにも考えていないところが“スナネズミ”に似てるといわれます。

かまた：大阪大学に合格したら，当チームが責任をもって，こき使ってあげます。

<Yさん(山口県)>1992年12月号の連載について質問。連載どおりに操作して，レンダリングしていたら,「ディスクの管理領域が破壊されています」と表示されます。どねーしたことでしょうか。さっぱりわかんねっす。CGAシステムのマニュアルにも載ってねー。うるうる。

かまた：うさ子さんはこのエラーメッセージの意味がわかりますか？

うさ子：ディスクの管理領域？　OSかなにかですか？

かまた：いいセンいってるけど，やっぱり知らんねんな。

うさ子：ごめんなさい。

かまた：いやいや，うさ子さんはPC-9801ユーザーやもん。これはディスクが壊れたときに出るX68000特有のエラーメッセージやねん。ディスクを交換すればいい。もちろん，CGAシステムのマニュアルにはないけど，X68000のマニュアルには載っているはず。

うさ子：でも，普通はパソコンのマニュアルを隅々まで読んだりしませんよね。

かまた：そう。知っていれば当たり前のことだけど，なかなか知る機会がないって，よくあるよね。

うさ子：皆さんも新学期からはコンピュータクラブに入ってみてはいかがですか？

<Yさん(福岡県)>マニュアルありがとうございました。おかげで僕もまっとうなパソコン人生（夜中にデバッグをしててニヤリと笑うような）が送れます（ニヤリ)。

かまた：まだまだアマイですね。当チームには，夢の中でデバッグをする者が数名おります。夜中の3時頃，突然ガバッと起きて，“××にバグを見つけた！”と叫びながらエディタを立ち上げ，数行書き換えてまた寝てしまうのです。

うさ子：まっとうなパソコン人生って，たいへんなんですね。

<Nさん（香川県)>以前のOh!XでPC-9801用RENDやそのほかを発表するって書いてあったけど，いつ頃になる予定ですか。ちなみに，どのくらい速くなるんですか？　全国の受験生のために，4月までには出してほしいな。

うさ子：2月号で紹介しましたように,「CGAマガジン創刊号」には，付録としてPC-9801用プログラムももれなくついています。

かまた：速度はPC-9801の機種によってかなり差がありますが，数倍は違うのではないでしょうか。4月から，ガンガン制作してくださいね。

<Sさん(東京)>FFEで，視点を(0,0,500)，注目点を(0,0,0)にすると，FFでエラーが発生する。注目点を(1,1,1)に変更しただけでエラーはなくなった。FFにバグがあると思われる。それだけ……。

うさ子：ちょっとやってみましょう。……FFやFFEでは別にエラーにはなりません（ただし，FFEの完成予想図には何も表示されない）が，RENDでエラーになりますね。「ターゲット指定がおかしい」と表示されますが，どういうことなんですか，かまたさん。

かまた：これはバグではなくて，仕様ですね。ちょっと難しい話になりますが，カメラワークというのは，視点と注目点と画角だけでは厳密には定まりません。画面の上方向を指定するパラメータが足りないのです。しかし，通常は画面の上方向は空間の上方向，つまりZ軸の＋方向に一致するために省略しているわけです。ところがこの例のように，視線がZ軸と完全に一致する場合，画面上にZ軸がなくなり，エラーが発生するわけです。

うさ子：先生，わかりませーん。

かまた：うっ，要するに，視線は真上を向いたり，真下を向いたりしたら，あかんちゅうこっちゃ。

<Sさん(東京都)>RENCON.Xが動きません。コマンドファイルや環境は完璧だと思うのですが，バグなどの情報は入っていませんか？　起動すると，コマンドファイルの中身の数行を表示して，何事もなかったように終わってしまいます。現在，手動レンコンしています。疲れたよ〜。それからBOMB.X。爆発の中心座標と，爆発させる物体との距離が，爆発の半径より長いときは影響がなくなると思っていたら，あるんですね。どういうことか教えてください。

かまた：わかりませーん。

うさ子：はいはい，BOMB.Xの作者であるMOOG寺田さんは，就職して千葉へ行ってしまったので，電話で聞いてみましょう。もしもし，寺田さんですか？

寺田：はい，寺田です。ただいま留守にしておりますので，ピーという発信音のあとに……。

うさ子：……ということで，寺田さんとは連絡が取れませんでした。ごめんなさい。

<Nさん(2月号の82ページと同じ人)>現在はRENDに頼りすぎている部分が目立つのも事実です。CGAシステムは今後も進化を続けていくでしょうが，どのような解答を出すのかがちょっと楽しみでもあります。

かまた：わかりませーん。

うさ子：こらこら，そんなことじゃいけません よ。地道に頑張っていきましょうね。

かまた：今年，CGA共通規格のフォーマット自体を改訂するための委員会が設置されます。ご意見などある方はお手紙ください。

た。このパーツはコーナーの内側にある赤白の縁石の直線部分です。縁石はコーナーで極端に内側を走ることを妨害するために設けられていますが，コーナーのちょっと手前の直線の部分から設定されています。この部分が「EDGE.SUF」です。たいていのカットではめったに使うことはないのですが，いま作っているカットでは15フレーム目あたりでコーナー用の縁石の中が空洞になっているのが見えています。そこで，この「EDGE.SUF」を置いてフタをしてやるというわけです。

EDGE.SUFの位置：(−5000,−1500,0)
Z軸回転　　　：180度

○後半の近景

「X02A.FSC」はこれでいいとして，「X03A.FSC」という名でセーブしてください。この調子で後半の「X03B.FSC」も作りましょう。

後半の近景は自由にやってみてください。私は以下のように配置してみました。参考にしてください。

SIBA.SUF
　位置：(　0,−1500,　0)
　位置：(2500,−4000,　0)
　位置：(5500,−4000,　0)
　位置：(5500,−6000,　0)
　位置：(5000,−2000,−5)
　位置：(2250,−1750,−5)
　拡大：X方向1.25倍，Y方向1.25倍
GUARD1.SUF
　位置：(6000,−3000,0)　Z軸回転：90度
　位置：(6000,−5000,0)　Z軸回転：90度
　位置：(6000,−7000,0)　Z軸回転：90度
MAKU.SUF
　位置：(4200,−4000,0)　Z軸回転：90度

横断幕「MAKU.SUF」を使用するときは，表裏に気をつけてください。これは「GUARD1.SUF」と違って，見てわからないようなことはありません。裏から見ると「Mandel 1」の文字が見えないからです。

最初はガードレールの位置に看板を並べてみたのですが，逆光で暗くなって画面が重苦しくなったのでやめました。でも，それだけだとものたりないので，横断幕を置いたわけです。

仕上げ

以上でいちおう完成してはいますが，より見栄えをよくするために，もうひとふんばりしてみましょう。

○道路にマッピングを行う

「ROADMAP.PIC」は道路に張りつけるタイヤの跡の画像です。「STRAI.SUF」や「R30.SUF」はマッピングに対応しておらず(UV座標がない)，「ROADMAP.PIC」を張りつけることができません。そこで，マッピングに対応した形状「M_＊.SUF」に取り替えてやります。

STRAI.SUF→M_STR.SUF
R30.SUF　→M_R30.SUF
R10.SUF　→M_R10.SUF

とはいってもFFEで物体を削除して，再び同じ位置に追加するのは非常に手間がかかります。こういうときは，エディタを使って直接ファイルに書き込んでしまえばいいのです。

ED X03A.FSC

で起動し，置換の機能を使って上記の3点を変更してください。

このとき，大文字と小文字の違いには十分注意してください。たとえば，

obj strai (: STRAI.SUF :)

というところを，

obj M_STR (: M_STR.SUF :)

としてはいけません。

obj m_str (: M_STR.SUF :)

としてください。

つまり，

obj ＜小文字＞ (: ＜どちらでもよい＞ :)

となっています(囲みの「ファイル名とオブジェクト名」参照)。

同様の処理を「X03B.FSC」にも行います。

○空気遠近法と有色光源

最後の隠し味として空気遠近法を使い，さらに光源にもちょっとだけ色をつけます。これもFFEを起動してすべての物体を読み込むより，エディタのほうが早いと思います。「X03A.FSC」の先頭付近を以下のように書き換

ファイル名とオブジェクト名

CGAのマニュアルのT-56ページ「補講：ファイル名とデータ名」とほとんど同じことです（形状ファイルのデータ名のことをオブジェクト名といっているだけ)。ファイル名とは，データが入っている箱につけた名前です。

つまり，

obj strai (: STRAI.SUF :)

ということは，“「STRAI.SUF」という名の箱に入っている「strai」という名の物体”ということを意味します。

ご存じのように，Human68kではファイル名は大文字でも小文字でも同じものとして扱います。しかし，CGAシステムのデータ名は大文字，小文字を区別します。つまり，「obj strai」と「obj STRAI」は別の物体になるのです。

では，オブジェクト名が大文字か小文字かはどこで決まるのかといえば，CADでセーブするときにどちらにするかで決まります。しかし，そんなことといちいち覚えていられませんので，特に理由がないかぎりオブジェクト名は小文字にすることにしましょう。

えてください。
(旧)

```
#frame( fno, 1, 25 )
@5.3@
fram
 {
        light pal( rgb ( 1.00 1.00 1.00 ) −2.00 −3.00 −4.00 )
```

(新)

```
#frame( fno, 1, 25 )
@5.3@
env  {  depth ( 9000 rgb ( 0.40 0.70 0.60 ) ) }
fram
 {
        light pal( rgb ( 0.80 0.90 1.00 ) −2.00 −3.00 −4.00 )
```

空気遠近法を設定する1行を加え，光源をやや青みがかった色にしています。カッコの個数などを間違えないようにしてください。

○バッチファイル

バッチファイルとは，コマンドラインから入力して実行する手順をファイルにしたものです。たとえば，「TEST.BAT」の中身が，

```
MKTCH TEST
HANIM TEST
```

ならば，「MKTCH TEST」を実行し，それが終了したあと，「HANIM TEST」を自動的に実行します(バッチファイルは拡張子を必ず「BAT」にしてください)。

作画を実行する「REND」のコマンドラインは非常に長くなりますが，1文字でも間違うと正しく動きません。そこでこれをバッチファイルにすると，確認や修正が楽になります。また，作画には非常に時間がかかりますので，普通は寝る前に実行させて，朝まで計算させておきます。しかし，今回のように「X03A」と「X03B」の2つに分かれている場合は，夜中に一度起きなければいけません。その点バッチファイルにしておくと，自動的に次のアニメーションを作画させることができます。「X03A」「X03B」の作画・アニメーション用のバッチファイルは「リスト1」のようになります。

数値演算プロセッサを持っている方は「REND」を「RENDXVI」に変更してください。また，X68000 XVI(要するに16MHz以上)でない方は，「CRD」を実行しないと滑らかなアニメーションにはなりません。その場合は「MKTCH」のあとを，

```
CRD X03¥X03A001 /OX03¥X03A
CRD X03¥X03B001 /OX03¥X03B
```

リスト1

```
MD X03
REND /A2 /G /HBACK¥BACKA001 /OX03¥X03A X03A.FSC M_STR.SUF M_R30.
SUF M_R10.SUF ROAD.ATR WILLI.* SIGN2.SUF SIGN6.SUF SIGN.ATR SIBA
.* EDGE.SUF GUARD*.*

REND /A2 /G /HBACK¥BACKB001 /OX03¥X03B X03B.FSC M_STR.SUF M_R30.
SUF M_R10.SUF ROAD.ATR WILLI.* BENET.* SIBA.* EDGE.SUF GUARD*.*
MAKU.*

MKTCH X03¥X03A001 X03¥X03B002
HANIM /M2 X03A
```

CGAマガジンバグ情報

臨時ニュースを申し上げます。本年1月中旬に発行されたばかりのCGAマガジン創刊号にバグが発見されました。症状は，「創刊記念特集 F1」の「5) カメラから走り去っていく」をハードディスクに展開できないようです。これに対して，関係者は「バグ緊急対策委員会」(taka2委員長)を設置し，以下のようなコメントを発表しました。

“エディタでディスク2の「¥BAT¥T_HDMK.BAT」の158行目に，「goto extract2」を追加してください”

(旧)

```
156: :next1
157: cd %work_dir%¥%animname%
158: for %%e in (%SOL%) do if not exist %%e goto extract2
159: lha e %SOURCE% %bat%
```

(新)

```
156: :next1
157: cd %work_dir%¥%animname%
158: goto extract2
159: for %%e in (%SOL%) do if not exist %%e goto extract2
160: lha e %SOURCE% %bat%
```

なお，同委員会はこの1行を加えることで，ほかのアニメーションが展開できなくなる可能性はほとんどないとみているものの，正確に判明するにはまだ数日かかるようです。

さらにTAKERUに登録しているCGAマガジンのマスターの修正も検討しており，もし順調に行えれば，2月以降にTAKERUで購入した分については，バグをなくせるとのことです。

今回の不祥事に対して，CGAマガジン編集長であるMax氏は，“おっかしいなあ。何回も動作確認したはずなんやけどなあ”，また発行人であるかまた氏は，“今回は誠に遺憾で，いかんなあ”と，いつものとおり，いいかげんなコメントを述べております。

HANIM X03A
としてください。

終わりに

いかがでしたか？ 思ったより簡単だったのではないでしょうか。このぐらいのCGAなら，慣れればサクサクできると思います。事実，CGAマガジンに入っている私が作ったカット（F1A～F1I）などは，雑事をこなしながら，1日1，2カットぐらいずつ制作しました。皆さんもぜひオリジナルカットを制作してみてください。

来月は“CGAマガジンの積極的な使い方”の最終回（上級編）として，CGAマガジンのカットを制作するときのテクニック，あるいは実際には使用するに至らなかったアイデアなどを紹介し，表現力をアップする方法について考えてみましょう。

さて，いよいよCGAコンテストです。今年も見応えのある作品が揃っています。しかし，いまのところは東京（新宿）と大阪（日本橋）でしか上映会が予定されていません。ほかの地域でも夏休みなどに上映会を行いたいのですが，どなたか会場を手配していただけないでしょうか。心当たりがありましたら，ご連絡ください。

DōGA 法人化への道

［これまでの話］

DōGAでは，ますます活発な活動を展開する半面，内部的な問題も大きくなっていた。その活動を支えるスタッフの大部分は学生のボランティアであるため，各自の活動には限界があり，一般的な会社組織と比較するとどうしても無責任でいいかげんなところが目立つ。面白くない雑用は誰もしない。

そのため，一部の責任感あるスタッフは非常に多忙で，単位を落として留年する者も少なくはなかった。現在の体制では，これ以上の活動は望めない。限界が見えてきたのだ。

そこでDōGAでは，数名の専任スタッフを常駐させることを検討する。そしてこの際，DōGA自身をちゃんと法人化することになった。

現在，DōGAは大学のクラブでもないし，会社でもない。法律上では存在しないのである。だから，税金も払っていないし，会社などと契約を結ぶこともできない（事務所はスタッフの個人名義で借りている)。ちゃんとした活動をするために，ちゃんとした組織になることから始めるわけだ。

しかし，法人とはいかなるものか？ 法律に詳しい者はおらず，諸々の手続きがわからない。さて，いったいどうなるのか？ スタッフは，理想と現実のギャップを目の当たりにすることになる。

＊　＊　＊

「これはあきらかに矛盾やないか！」
「そのとおり。しかし，そんなことをいっても問題は解決しない」

法人には株式会社，有限会社，財団法人などがある。前の2つが営利目的であるのに対して，財団法人は“科学，文化などに貢献するための非営利団体”である。だから，DōGAも財団法人を目指していた。しかし……。

「DōGAの目的は，CGAという新しい映像文化の普及や。だから当然，財団法人やないか。なのに，実際の手続き上，資本金は数億円で，大臣か知事かのあと押しがいるって？」
「私に文句をいっても困るよ。お役所がそういっているんだから」
「じゃ，どうしろっていうんだ？」
「結局，法人化するなら事実上株式会社か有限会社かしか選択の余地はないそうだ」
「うん，読者からもその方面に詳しい人からいろいろお手紙もらったけど，やっぱりそれしかないみたい。“DōGA生活共同組合案”なんかもあったけど，実際検討してみるとやっぱり無理があるみたい」
「でも，株式，有限どちらにしろ，営利目的の団体やないか」
「でも，それしかできへんのなら，しゃーないやん」
「こう考えたらどうや。法人化は目的やない。専任スタッフを置いて，しっかりした活動をするのが目的や。要するに，法人化は単なる形式やねん。たとえ株式会社になっても，DōGAの目的は変わらへんし，いままでどおりの活動を続けたらええんやないか」
「そうはいっても，株式会社になったら外部の人は営利団体やと思うで」
「実際の活動を行っていく過程で誤解を解いていけばいいやん」
「それだけで十分とは思えんけど」
「DōGA自身は従来どおりアマチュアの団体として残して，それを援助する雑用会社を別に作るって形にすれば？」
「オレは，“形式”はどんな“形式”でもかまわんで」
「たしかに別会社ならわかりやすいな。専任スタッフは置けるし，対外的に契約なんかが必要なときはその会社にやってもらえばいいし」
「それって，幽霊会社っていわへん？」
「さあ？ よう知らんけど」
「みんな非営利にこだわってるけど，実際問題として活動を続けていくには資金が必要なんやで。どうすんの？ 営利団体の株式会社でええんとちゃうの？」
「誤解のないように。たとえ財団法人でも，収入はあっていいんやで。目的や利益の還元先が違うねん」
「難しいことはわからんけど。参加，協力してもらっている一般ユーザー相手に営利目的の行為をするのは好かん。だから，営利行為をしている一般企業から映像制作の仕事なんかを受けて，それを資金に充てて，一般ユーザーには非営利活動を続けたらええと思う」
「なんかネズミ小僧みたい（笑）」
「たしかにDōGAの目的や活動って，ある種の“運動”だから，金銭抜きに賛同して初めて意義があると思う」
「それって，“宗教”やで（爆笑）」
「そういえば，知人の会社がDōGAのことに興味をもってて，仕事を発注したいっていうてるんやけど，有限会社やと発注できへんねんて。ちゃんと株式会社にしといてっていわれたわ」
「そうそう，有限会社やと事務所借りられへんということもあるらしいで」
「じゃあ，株式しかないな」
「だれが株主になるん？ 下手に外部の人を株主にすると，いろいろ口出されたりしてややこしいで。株主にはそういう権利があるからな」
「いやゃな。DōGAには株主はいらんねんけど，株主のない株式会社って無理やろな」
「当たり前やがな」
「アマチュアの団体であるDōGA自身が株主になるとか，大阪大学コンピュータクラブ，京大マイコンクラブとか」
「そんな法律上存在していない団体が株主になれるやろか」
「う～ん，法律に詳しいヤツはおらんのかな」
「ここから先はプロがおらんと話進まんな」

ということで，とりあえず税理士さんのところへ相談に行くことになった。この調子では，いつ，どのような形式で法人化するかはわからない。現在のところ，“4月から株式会社”案が有望。そういえば，1992年7月号では“株式会社だけにはしない”って書いたような気がするが，別にウソを書くつもりはなかった。上記のような事情を理解していただきたい。どうしても株式会社には反対で，それなら賛同できないから，オレが出した分のカンパを返せという方がいたら，それに応じるつもりなので申し出てほしい。

このように，設立の手続きは遅れているが，実務面での準備は少しずつ話がまとまってきた。設立時の専任スタッフは3名。ひとりはこの春大学をやめ，ほかの2人は勤めていた会社をやめる手続きを取った。もうあとへは引けない。さあ，いったいどうなるんだ？ 次回，続編を待て。

各種ツールを使ったモデリング(3)

文月　涼

■ごめんなさい

実は気づいていたのですが，Oh!Xの付録ディスクに入っていたTUBE.Xは開発中のもので，私の手元にあるバージョンとは異なります。このため記事どおりに処理していても，うまくいかない可能性があることが判明しました。てっきりDōGAがなんらかのフォローを行っていると思っていたのですが，どうやら何もやっていなかった模様です。大手のネットには差分ファイルをアップしたものの，一般の方へのフォローは終わっていません。「CGAマガジン」には入れるはずだったのですが，度重なる担当者の敵前逃亡の大混乱のなか，うやむやになってしまったようです。なんとかフォローする方法を考えます。ごめんなさい。

■先月のおさらい

さて，先月のおさらいとして，実際にCADで断面図を打ち込み，TUBEで断面図を作り，KAMAで合成する実習をやってみましょう。

サンプルとして提供する断面のデータをCADで実際に打ち込んでみてください。CADを立ち上げて，Wキーを押してX座標を入力し，Eキーを押してY座標を入れ，Rキーを押してZ座標を入れて，スペースキーで点の座標を決定します。これを繰り返し，サンプルの面を入力してください。先頭からズンタカ打ち込んでいくと，数面はそのまま打ち込めるのですが，たぶん5面目でCADにピッと怒られるでしょう。当然ですね。単一の面を構成する点が同一平面上にないのですから。

こういったときはどうすればいいのかと申しますと，例の「エディタがりがり」でデータを作成するのです。

とりあえず現在まで入力した面をセーブして，エディタで対象のSUFファイルを読み込みます。そして，そのファイルの最終行の「}」の前に，面を手で書き込んでいくのです。各座標と座標の間はTABなりで，適当に仕切ってタッタカ打ち込んでみてください。SUFファイルの文法，各単語の意味を理解したい人は，CGAシステムのマニュアルのCGA共通企画を熟読しましょう。

TUBEにかける断面図

完成した胴体のオブジェクト

入力が終了したら，ファイルをセーブ後，もう一度対象ファイルをCADで読み込んでください。どうでしょう。すべての点が同一平面上になかった面でも，ちゃんと読み込まれますね。つまり，CADにおける面の面としての正当性のチェックは，データ入力時にしかなされていないのです。したがって，外部で作った不正な面はCAD上でその面自身をトレースでもしないかぎり，読み書き自由なのです。

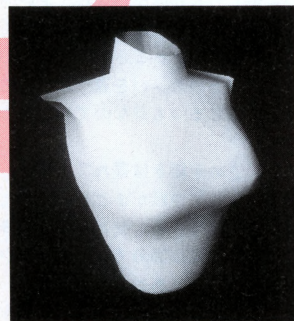
では，実際に私がこの断面図を入力したときはどうしたのでしょうか。私の場合，まずすべての面を平面としてCADで入力し，そのうえでエディタで各点を上下させ，再度CADで雰囲気をチェックしています。いちいちCADを終了していると時間がかかるので，CADのファイルメニューのCOMMANDからCOMMAND.Xを起動し，エディタでSUFを編集して，再度CADに戻り，SUFを再び読み込んでいます。

課題として提供した断面は，女性の人体モデルを作る段階で使用した上半身のそのまた半分です。鋭い人はこの断面を見た段階でいくつかの点に気づくでしょう。まずひとつは「左右対称の物体であれば半分を作って，もう半分は合成する」という手法です。次には「各断面図の角数が同じ」です。断面図の面積に著しい差があっても，必ず同じ角数で作っています。何度もお伝えするようですが，TUBEは決して角数が同じでないと処理できないのではなく，角数を同じにしたほうがよりきれいに処理できるのです。これらの点を念頭において断面図を作るといいでしょう。

■TUBEする

次にこの断面図をTUBEにかけます。蓋を閉じないオプションをつけて処理してください。すると上半身の半分の筒ができるはずです。このままKAMAでもう半分を合成してもいいのですが，それでは次のSHADEで困ってしまうので，このSUFファイル(DOC.SUFとする）をエディタに読み込んで処理します（SHADEをかける段階で法線ベクトルがおかしくなってしまうため）。DōGAの物体はデフォルトでX軸のマイナス方向を向いているので，胴体の中央の平面はY＝0の面となります。

ファイルを読み込んだら上から順に面を見ていって，Y＝0の面を見つけたら（三角形が続いて2面Y＝0であるはず）その面をエディタで削除します。ファイルの最終行まで削除し終えたら，編集ファイルの名前を変更してセーブします（DOD.SUFとする）。

次にFFEを立ち上げ，DOD.SUFを読み込んで決定，もう一度DOD.SUFを読み込み，Yの倍率を－1にして，もう一度決定。そしてファイルをセーブします(DOE.FSC)。

DOE.FSCをKAMAって，DOE.SUFを生成し，DOE.SUFをSHADEにかけ，出力ファイル名をDOF.SUFとして出力します。このDOF.SUFをAUTOで“-A2 -G”のオプションをつけて実行すると，女性の上半身のモデルが回転する画像が生成されるはずです。

これでひととおりTUBEを使ったプロセスを体験したので，あとは応用していろいろ試してみてください。

TUBEを使ったモデリングは，物体をどういう断面で捉えるかが勝負です。またその断面は必ずしも平面である必要はないのです。たとえば，今回の例では上半身だけを作ってみましたが，全身の断面図を先に作成しておいて，断面図を動かす物体ごとに切り分け，おのおのTUBEで造形することもできるはずです。

そして，前回の「分割の要のラインとなるラインを決めておく」という話を忘れずに。人体モデルであれば，胴体の断面図を腰と胸に分割したときも，ウエストのラインをいじらないでおけば，あとで人体モデルを組み上げたときに，少なくとも直立している状態では，よどみのない美しいボディラインが再現されるのです。

■パーツのチェックと合成

車などの複雑な物体を別々にモデリングしていくと，最終的には車の形に組み上げなければなりません。その組み上げ方（合成）にも２種類あり，それが固定合成と動的合成（仮につけた名前）です。

固定合成とは車の本体にバンパーをつけるように，後々物体を個々に動かす必要がない合成です。動的合成とは人体モデルのように，物体を一定の法則に従って動かす必要がある合成を意味します。

どちらの場合でも作成しておいた図面がものをいいます。複数の物体を固定合成する場合，いきなり画面を見ながらCADやFFEで手探りでもできるのですが，やはり正確な座標を図面上で押さえておいて，移動距離を一発入力したほうが無難です。

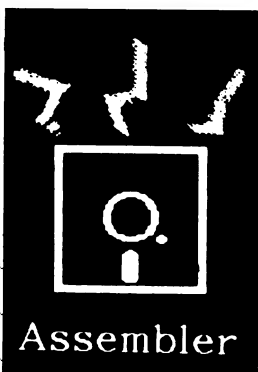
また動的合成は物体としては合成せずに，レンダリング時のフレームソースでその動きの法則性を規定します。この場合でも，きちんとした座標がわかっていないとつらいものがあります。

動的合成についてはマニュアルの構造体の章に譲るとして，固定合成を説明します。固定合成はFFE＋KAMAを用いる方法が無難です。それぞれ別体としてモデリングしておいた物体を次々にFFEに読み込んで，あるべき位置に配置し，そのFSCファイルをKAMAで合成します。

しかし，FFEは微調整をしようとしても物体の読み込みが遅いため，一度FSCを作成したら，あとの微調整は直接FSCファイルを書き換えると便利です。KAMAはFSCの文法をくずさなければまったく問題はありません。

ページがなくなってしまったので，ごめんなさい，また来月です。

リスト1

```
obj suf harab (
atr dan
prim poly (       80        0         -30
      78       -18      -30
      76       -30      -30
      71       -47      -30
      60       -72      -30
      38       -90      -30
      0        -96      -30
      -28      -92      -30
      -48      -84      -30
      -64      -72      -30
      -74      -54      -30
      -79      -20      -30
      -76      0        -30 )
atr mune
prim poly (        100       0         0
        98       -23      0
        96       -38      0
        89       -59      0
        75       -91      0
        48       -113     0
        0        -120     0
        -35      -115     0
        -61      -106     0
        -80      -91      0
        -93      -68      0
        -99      -26      0
        -96      0        0 )
prim poly (  108       0         50
       106      -25      50
       104      -41      50
       96       -64      50
       81       -99      50
       51       -123     50
       -2       -130     50
       -40      -125     50
       -69      -115     50
       -90      -99      50
       -105     -74      50
       -111     -28      50
       -108     0        50 )
prim poly ( 116      0         100
      109      -49      100
      95       -87      100
      81       -113     100
      64       -133     100
      40       -142     100
      0        -146     100
      -38      -145     100
      -73      -136     100
      -103     -113     100
      -124     -68      100
      -129     -30      100
      -123     0        100 )
prim poly (        121       0         150
      113      -52      150
      98       -93      150
      83       -122     150
      65       -143     150
      39       -153     150
      -4       -157     150
      -45      -156     150
      -82      -116     150
      -115     -122     150
      -137     -73      150
      -143     -32      150
      -136     0        150 )
prim poly (        123       0         195
      117      -46      192
      109      -90      188
      100      -117     192
      80       -150     195
      38       -166     200
      0        -170     200
      -43      -167     200
      -90      -162     200
      -125     -139     200
      -147     -105     200
      -159     -45      200
      -152     0        200 )
atr hara
prim poly (        153       0         235
      155      -60      235
      158      -100     235
      152      -135     235
      126      -157     235
      82       -171     235
      0        -180     235
      -61      -180     235
      -102     -173     235
      -125     -162     235
      -155     -132     235
      -169     -38      235
      -164     0        235 )
prim poly (        162       0         280
      165      -60      280
      169      -100     280
      159      -140     280
      129      -167     280
      78       -182     280
      0        -188     280
      -66      -190     280
      -111     -178     280
      -138     -166     280
      -158     -138     280
      -175     -41      280
      -170     0        280 )
prim poly (        125       0         335
      126      -51      335
      128      -95      335
      123      -124     335
      99       -155     335
      60       -177     335
      0        -190     335
      -61      -189     335
      -108     -184     335
      -139     -172     335
      -157     -143     335
      -171     -42      335
      -170     0        335 )
prim poly (        72        0         400
      68       -49      400
      50       -111     400
      39       -148     400
      17       -170     400
      0        -185     400
      -35      -189     100
      -97      -181     100
      -124     -176     400
      -141     -158     100
      -155     -104     100
      -161     -35      400
      -158     0        100 )
prim poly (        10        0         115
      33       -58      115
      23       -107     445
      13       -112     415
      1        -166     445
      -25      -218     415
      -59      -235     445
      -99      -231     415
      -125     -195     115
      -131     -135     415
      -113     -66      445
      -146     -23      145
      -113     0        415 )
prim poly (        20        0         162
      17       -18      165
      6        -48      470
      -6       -57      175
      -21      -61      480
      -41      -65      182
      -55      -66      483
      -86      -69      480
      -117     -69      470
      -132     -57      465
      -140     -39      463
      -142     -19      462
      -140     0        462 )
prim poly (        6         0         515
      3        -14      516
      -5       -38      518
      -14      -45      521
      -27      -48      521
      -43      -52      529
      -55      -52      532
      -80      -55      538
      -106     -55      545
      -118     -45      519
      -124     -31      550
      -126     -15      551
      -124     0        550 )
)
/* filename = DOB.SUF */
/* オブジェクト名 harab */
/* オブジェクト数 1 */
/* ポリゴン数 13 */
/* 次のオブジェクト 0 */
```

文字列照合アルゴリズム

Murata Toshiyuki 村田 敏幸

ワープロやエディタなどでひんぱんに活用されている検索機能。今月はそこで用いられるような，文字列のなかからある一定の文字列を探し出すアルゴリズムを考えてみましょう。単純なアルゴリズムに加えて，さらに洗練された2つの方法を紹介します。

今回は文字列照合(string matching)を取り上げる。2つの文字列textとpatternが与えられたときに，patternと一致する部分文字列をtext中から探す処理だ。文字列探索/検索(string searching)といったほうが通りはよいかもしれない。FIND.Xのような文字列検索ツールや，エディタ/ワープロの検索機能を思い浮かべてもらえばよい。

単純なアルゴリズム

文字列照合を実現するわかりきった方法は，被照合側文字列（以下「テキスト」と呼ぶ）のすべての文字位置から始まる個々の部分文字列と，照合パターン文字列（以下，単に「パターン」と呼ぶ）を力まかせに比べていくことだ。まず，テキストとパターンの先頭を揃えて重ね，頭から1文字ずつ比較していく。パターンの末尾まで一致したら照合は成功，途中で不一致が検出されたらパターンを1文字分右(テキスト末尾方向)にずらして，また頭から比較する。この様子を図1に示した。図1では，比較の結果一致した文字は下線つきで，不一致だった文字は四角で囲んで表してある。何の印もついていない文字は比較されなかった文字だ。

文字列照合の場合，テキストをすべて走査し終わるまでに行う文字比較の回数がアルゴリズムの性能の目安になる。テキストがn文字，パターンがm文字で，nがmに比べて十分大きいとすると，いま示した単純法では最悪m×n回近くの文字比較を必要とする。単一の文字で埋まったテキスト中から，その同じ文字の並びに1文字だけ異なる文字をつけ足したパターンを探す場合がその最悪のケースとなる。たとえば，テキスト“AA……A”からパターン“AA……AB”を探す場合だ。この場合，m－1文字まで一致し，最後のm文字目で不一致が検出される，というm文字分の比較がn回近く繰り返されるわけだ。

もっとも，現実には，このような場面は滅多に現れないといってよい。仮に各文字がテキスト中に現れる確率が等しいとすると，テキスト中の1文字とパターン中の1文字が一致する確率は「文字の種類の逆数」で，2文字，3文字と連続して一致する確率はその2乗，3乗となる。文字の使用頻度は文字ごとに異なるため，このような単純な確率論は正確さに欠けるとはいえ，現実の文字の種類は十分多いから，2つの文字が一致する確率よりも一致しない確率のほうがずっと高いことだけは間違いない。何文字も連続して一致する確率ともなれば無視しうるほど小さい。単純文字列照合アルゴリズムでは，早めに不一致が検出されるほど素早くパターンをずらして核心に迫っていけるわけであり，つねにパターン先頭で不一致が見つかるとするならテキストを走査し終わるまでにn文字の比較しか行わずに済む。そして，確率上，それに近い状況は結構頻繁に起こることがわかる。単純法における平均の文字比較回数は，nよりもそれほど多くはないということだ。このため，平均の実行時間も，想定される最悪のケースよりはずっと短く，m×nというよりむしろnに比例する程度に収まると考えられる。

一般に，実行時間がデータの量にのみ比例するアルゴリズムは「高速」，少なくとも「実用的」な部類には属する。単純な文字列照合法は，実用上，十分高速なのだった。これは，いわば無策の勝利であり，文字列照合の分野には複雑なアルゴリズムの入り込む余地があまりないことを意味する。歴史的には，それゆえ，より高速な文字列照合アルゴリズムの登場がずいぶん遅れたという経緯がある[1]。昔話はともかくとしても，文字列照合アルゴリズムを考案/改良/実装する際には，シンプルさを保たないと単純な方法には勝てないということは頭に入れておきたい。

では，単純法による文字列照合サブルーチンの例をリスト1に示そう。リスト1のサブルーチンstrstrはテキストとパターンの先頭アドレスをスタックに積んで呼び出すと，照合に成功した位置をa0に，Z＝1をccrに返す。失敗した場合，a0は不定で，Z＝0が返る。今回作成するサブルーチンの戻り値はすべてこの形に統一してある。

1）ソーティングの分野では，1962年の時点ですでにクイックソートの論文が発表されているが，文字列照合アルゴリズムが発表されたのは1977年だ（考案はもう少し前）。

プログラムは，23～26行のループでパターン先頭の文字を探し，見つけたら30～35行のループで残りを比較するという構成になっている。特に強調するほどのことではないが，パターン先頭の文字を特別扱いして，つねにレジスタに保持しておくことで，無駄なメモリアクセスを減らしている点を指摘しておく。

わずかな変更を加えることにより，リスト1の平均性能はもうすこし向上する。35～36行では，34行の文字比較の結果が不一致だったらラベルloop1へ，一致したらラベルloop2へ分岐することになるが，この分岐の順序を逆にして，

```
        bne     loop1
        bra     loop2
```

とすればよい。不一致の確率のほうが一致する確率より高いことを考慮して，分岐が起こりやすいほうを先に持ってくるわけだ。

また，テキスト末尾に達したかどうかのテストを23～26行のループと，30～35行のループの2カ所でしていることに目をつけると，2度目のテストは省略できることがわかる。30～35行は，

```
loop2:  move.b  (a3)+,d0
        beq     match
        cmp.b   (a2)+,d0
        beq     loop2
```

のように簡略化してしまってよい。この場合，テキスト末尾の検出がワンテンポ遅れることになるために本来しなくてもよかった文字比較を行う可能性が出てくるが，パターンよりもテキストが十分長ければループ中から無駄な命令を追い出したことによって浮く時間のほうが大きい。

もっと大がかりな改造案としては，事前にパターンの文字数を数えておき，ループをdbra系の命令で構成することで，30～35行のループからパターン末尾のテストも追い出すというアイデアが考えられる。もっとも，このあたりまでくると，効果のほどは微妙だ。先に挙げた2つの改良案も結局はそうなのだが，効率が改善されるのは第2のループだけであり，実行時間の大半が第1のループで費やされることを考えると，性能が上がるとしてもごくわずか。文字列の長さを数える手間がその分を相殺してしまうかもしれない。確かに，テキストが十分長ければ文字数を数える時間は無視できるので，元のリスト1より遅くなることだけはないだろうが，上の2つの改良のみを加えた版との優劣はよくわからない。ま，ごたくを並べていても始まらないから，とりあえず手を入れるだけ入れて，あとで実行速度を比較してみることにしよう。

リスト2がそのパターン長事前計数版だ[2)]。42～43行はふつうの感覚だと，ループの中身を軽くするために，

```
        dbne    d4,loop2
        bne     loop1
```

2）リスト2ではdbraを利用する都合でパターン長が65,536文字に制限されていることに注意。特殊な応用でこれが問題になる場合には，もうひとつdbraを使ってループを二重にしなければならない。

図1　単純な文字列照合法

```
 1)  TOKKYOKYOKAKYOKU  ←テキスト
     [K]YOKU           ←パターン

 2)  TOKKYOKYOKAKYOKU
      [K]YOKU

 3)  TOKKYOKYOKAKYOKU
       K[Y]OKU

 4)  TOKKYOKYOKAKYOKU
        KYOK[U]

 5)  TOKKYOKYOKAKYOKU
         [K]YOKU

 6)  TOKKYOKYOKAKYOKU
          [K]YOKU

 7)  TOKKYOKYOKAKYOKU
           KYOK[U]

 8)  TOKKYOKYOKAKYOKU
            [K]YOKU

 9)  TOKKYOKYOKAKYOKU
             [K]YOKU

10)  TOKKYOKYOKAKYOKU
              K[Y]OKU

11)  TOKKYOKYOKAKYOKU
               [K]YOKU

12)  TOKKYOKYOKAKYOKU  照合成功
                KYOKU
```

リスト1　STRSTR.S

```
 1: *           単純な文字列照台
 2:
 3:             .xdef    strstr
 4: *
 5:             .offset 4
 6: str:        .ds.l    1                  *テキスト
 7: pat:        .ds.l    1                  *パターン
 8: *
 9:             .text
10:             .even
11: *
12: strstr:
13: SAVREGS =            d0-d2/a1-a3
14: SAVSIZ  =            (3+3)*4
15:             movem.l SAVREGS,-(sp)
16:
17:             movem.l str+SAVSIZ(sp),a0/a1
18:                                         *a0 = テキスト
19:                                         *a1 = パターン
20:             move.b   (a1)+,d2           *d2 = パターン先頭文字
21:             beq      retn               *パターンが空文字列だった
22:
23: loop1:      move.b   (a0)+,d0           *d0 = 入力文字
24:             beq      nmatch             *見つからなかった
25:             cmp.b    d2,d0              *パターン先頭文字に出会うまで
26:             bne      loop1              *  ポインタを進める
27:
28:             movea.l a1,a3               *パターン2文字目以降と
29:             movea.l a0,a2               *  順に比較する
30: loop2:      move.b   (a3)+,d1           *
31:             beq      match              *
32:             move.b   (a2)+,d0           *
33:             beq      nmatch             *
34:             cmp.b    d1,d0              *
35:             beq      loop2              *
36:             bra      loop1              *不一致だったのでやり直す
37:
38: nmatch:     moveq.l #-1,d0              *見つからなかった(Z=0,N=1)
39:             bra      retn
40:
41: match:      subq.l   #1,a0              *見つかった(Z=1,N=0)
42:                                         *a0 = その位置
43: retn:       movem.l (sp)+,SAVREGS
44:             rts
45:
46:             .end
```

としたくなるが，例によって，どちらの分岐が起こりやすいか，という考え方をすると，確率上，リスト2のほうが効率的だという結論になる。

さて，効率はともかく，リスト1，2には実用上の欠陥がある。1文字＝1バイトを仮定しているために，全角文字交じりの日本語テキストを与えると誤動作することがあるのだ。たとえば，パターンとして半角の“A”（文字コード41_H)を与えたときに，全角カタカナの“ア”のシフトJISコードが8341_Hであるために，下位バイトが引っ掛かってリスト1，2は誤った結果を返す。いまはプログラムが複雑になるのを嫌ってこのまま放っておくが，いちおう，対策だけは考えておこう。

この問題の一般的な解決法は，テキスト，パターンを先頭から走査していく過程でつねにシフトJIS漢字コードの第1バイトを意識し，見つかったら続く1バイトも一緒に取り出して，比較を確実に“文字単位”で行うようにすることだ。ただ，実行速度がかなり落ちるので，なるべくならこの方法は使いたくない。

別の案としては，文字列を8ビットデータの配列ではなく，16ビットデータの配列で表現するという方法がある。1バイトコードの上位に00_Hを補って，すべての文字を2バイトで表すわけだ。こうすれば，1バイト文字と2バイト文字を区別しなくても済むため，事実上，1バイト文字だけを扱うのと変わらなくなる。日本語交じりのテキストを扱う場合には，文字列をこの形式に統一すると，照合以外の文字列操作も楽になることが多い。メモリを余分に使うことにはなるが，考慮に価する選択肢といえる。

あるいは，とりあえず照合はバイト単位で行い，照合が得られてから，その位置が2バイト文字の途中から始まっていないかどうか検査するという方法も考えられる。一般に，シフトJIS漢字コードを含む文字列から途中の1バイトだけを取り出しても，それが，1バイト文字なのか，2バイト文字の上位バイトなのか，下位バイトなのかは区別できないから，この方法は成り立たないようにみえるが，やりようもないではない。のちほど，具体的な方法を示すことにする。

リスト2　STRSTR2.S

```
 1: *         単純な文字列照合（別版）
 2:
 3:           .xdef   strstr
 4: *
 5:           .offset 4
 6: txt:      .ds.l   1                   *テキスト
 7: pat:      .ds.l   1                   *パターン
 8: *
 9:           .text
10:           .even
11: *
12: strstr:
13: SAVREGS  =        d0-d4/a1-a3
14: SAVSIZ   =        (5+3)*4
15:           movem.l SAVREGS,-(sp)
16:
17:           movem.l txt+SAVSIZ(sp),a0/a1
18:                                       *a0 = テキスト
19:                                       *a1 = パターン
20:           move.b  (a1)+,d2            *d2 = パターン先頭文字
21:           beq     retn                *パターンが空文字列だった
22:
23:           movea.l a1,a2
24: loop0:    tst.b   (a2)+
25:           bne     loop0
26:           subq.l  #1+1,a2             *先頭除外の分とdbraの分
27:           suba.l  a1,a2
28:           move.w  a2,d3               *d3 = パターン長-1のdbraカウンタ
29:
30: loop1:    move.b  (a0)+,d0            *終端に達するか
31:           beq     nmatch              *  パターン先頭の文字と
32:           cmp.b   d2,d0               *  一致する文字が見つかるまで
33:           bne     loop1               *  ポインタを進める
34:
35:           move.w  d3,d4
36:           bmi     match
37:
38:           movea.l a1,a3               *パターン2文字目以降と
39:           movea.l a0,a2               *  順に比較する
40:           move.w  d3,d4               *
41: loop2:    cmpm.b  (a2)+,(a3)+         *
42:           bne     loop1               *不一致だった
43:           dbra    d4,loop2            *
44:
45: match:    subq.l  #1,a0               *見つかった(Z=1,N=0)
46:                                       *a0 = その位置
47:           moveq.l #0,d0
48: retn:     movem.l (sp)+,SAVREGS
49:           rts
50:
51: nmatch:   moveq.l #-1,d0              *見つからなかった(Z=0,N=1)
52:           bra     retn
53:
54:           .end
```

クヌース-モリス-プラット法

単純な文字列照合アルゴリズムでは不一致が検出されるとパターンを1文字分ずらしてふたたび照合を試みた。より洗練されたアルゴリズムでは，このパターンの移動量をなるべく大きくすることで効率を稼ごうとする。そのようなアルゴリズムのひとつに，KnuthとPratt，および，彼らとは別にMorrisが同時期に考案したKnuth-Morris-Pratt(KMP)法がある。

図2を見てもらおう。先ほどの図2と同じ例にKMP法を適用した場合の比較の様子を示してある。最初の数ステップは単純なアルゴリズムと同じ動きをしているが，図2の4)から5)，5)から6)ではパターンが一気に3文字ずれていることに注目しよう。4)と5)は，パターンの5文字目で不一致が検出された場面，逆のいい方をすれば，4文字目までは一致した場面だ。KMP法ではこの一致した部分文字列か

図2　KMP法による文字列照合

```
1) TOKKYOKYOKAKYOKU  ←テキスト
   KYOKU              ←パターン

2) TOKKYOKYOKAKYOKU
    KYOKU

3) TOKKYOKYOKAKYOKU
     KYOKU

4) TOKKYOKYOKAKYOKU
      KYOKU

5) TOKKYOKYOKAKYOKU
         KYOKU

6) TOKKYOKYOKAKYOKU
            KYOKU

7) TOKKYOKYOKAKYOKU
             KYOKU

8) TOKKYOKYOKAKYOKU  照合成功
              KYOKU
```

ら情報を引き出す。

図2の例の場合，一致したのは“KYOK”の4文字だ。この文字列を，適当にずらしながら重ねてみると，

　　KYOK？
　　　KYOK

のように1文字ずらしただけでは，「？」のところにどんな文字を置いたところで絶対に照合は得られない。2文字ずらして，

　　KYOK？？
　　　　KYOK

でもだめ。つぎのように3文字ずらし，一致した部分の先頭と末尾の“K”を重ねて初めて一致する可能性が出てくる。

　　KYOK？？？
　　　　　KYOK

ここで，どれだけずらせば照合が得られそうかを毎回調べるのでは単純な文字列照合アルゴリズムと同じになってしまう。しかし，この移動量はパターンによってのみ決まるから，あらかじめ，何文字目で不一致が見つかったらパターンをどれだけずらすか，という形でテーブルにしておくことができる。同じパターンで何度も文字列照合を試みる場合でも，このテーブルは事前に1回作成しておけばよい。例のパターンの場合，各文字位置でのパターン移動量は図3のように決まる。なお，図3では示されていないが，パターン先頭との比較で不一致が見つかった場合は，無条件にパターンを1文字分ずらすのはいうまでもない。

このように，パターン先頭部との部分的な一致が生じたときに，その情報を基に大きくパターンを移動する，というのがKMP法の骨子だ。では，もうすこし細部を検討していこう。

図2に戻って，パターンを3文字移動したあとで比較をやり直す場面，5)と6)を見てほしい。パターン先頭の“K”に下線がついていないのは誤りではない。この“K”はテキストとは比較されないのだ。

図3　KMP法のパターン移動量

パターン＝KYOKUの場合

2文字目で不一致の場合　K？
1文字ずらせば一致の可能性あり　K

3文字目で不一致の場合　KY？？
1文字ずらしても一致しない　KY
2文字ずらせば一致の可能性あり　KY

4文字目で不一致の場合　KYO？？？
1文字ずらしても一致しない　KYO
2文字ずらしても一致しない　KYO
3文字ずらせば一致の可能性あり　KYO

5文字目で不一致の場合　KYOK？？？
1文字ずらしても一致しない　KYOK
2文字ずらしても一致しない　KYOK
3文字ずらせば一致の可能性あり　KYOK

直前のステップでテキストとパターンの“K”が重なるようしたのだから，比較しなくても一致するのはわかっている。したがって，パターン移動後の再比較はパターンの2文字目から行えばよい。当然，こうやって比較を省略するためには，パターンの何文字目から再比較を行うか，という情報が必要になる。しかし，図2を注意深くみると，「1文字以上一致したあと」に限れば，不一致が検出されたときの再比較はその不一致を起こした位置から始まっていることがわかる。たとえば，4)で不一致が見つかったテキスト上の位置と，5)で再比較を始める位置は同じであり，この関係は，

　　3)と4)
　　5)と6)
　　6)と7)

でも成立している。このため，パターンをどれだけ

リスト3　KMP.C

```
 1: /*  KMP法による文字列照合   */
 2:
 3: #define NULL     ((void *)0)
 4: #include    <string.h>
 5:
 6: #define MAXLEN  256
 7:
 8: static int kmp_len;
 9: static unsigned char kmp_pat[MAXLEN];
10: static unsigned char kmp_table[MAXLEN];
11:
12: int kmp_comp(unsigned char *pat)
13: {
14:     int i, j;
15:
16:     if ((kmp_len = strlen(pat)) >= MAXLEN)
17:         return -1;
18:
19:     strcpy(kmp_pat, pat);
20:
21:     for (kmp_table[i = 1] = (j = 0); pat[i] != '¥0';) {
22:         if (pat[i] == pat[j]) {
23:             kmp_table[++i] = ++j;
24:         } else if (j == 0)
25:             kmp_table[++i] = 0;
26:         else
27:             j = kmp_table[j];
28:     }
29:     return kmp_len;
30: }
31:
32: unsigned char *kmp_exec(unsigned char *text)
33: {
34:     int i, j;
35:
36:     if (kmp_len == 0)
37:         return text;
38:
39:     for (i = j = 0; text[i] != '¥0';) {
40:         if (kmp_pat[j] == '¥0')
41:             return &text[i - kmp_len];
42:         else if (text[i] == kmp_pat[j]) {
43:             i++;
44:             j++;
45:         } else if (j == 0)
46:             i++;
47:         else
48:             j = kmp_table[j];
49:     }
50:     return NULL;
51: }
52:
53: #ifdef  MAIN
54:
55: #include    <stdio.h>
56: #include    <stdlib.h>
57:
58: void main(int argc, unsigned char **argv)
59: {
60:     int n;
61:     unsigned char buff[256], *p;
62:
63:     if (argc < 2)
64:         exit(EXIT_FAILURE);
65:
66:     n = kmp_comp(*++argv);
67:     while (fgets(buff, 256, stdin) != NULL)
68:         if ((p = kmp_exec(buff)) != NULL)
69:             printf("%.*s¥x1b[31m%s¥x1b[m%s",
70:                 p - buff, buff, *argv, p + n);
71:
72:     exit(EXIT_SUCCESS);
73: }
74:
75: #endif
```

ずらすかが決まれば，再比較を始めるパターン上の位置も自動的に決まる。パターンをずらしたあとで，直前に不一致が見つかったテキスト中の文字と重なる位置，それが再比較の開始位置となる。実際のプログラムでは，パターンの移動量よりも再比較を始める位置が直接わかったほうが都合がよいから，ふつう，こちらについてのテーブルを用意する。

肝心のテーブルの作成に関しては，巧妙な方法がある。ここでやりたいのはパターン自身をパターンの第2文字目以降と照合することにほかならないので，KMP法そのものを適用し，その中間結果を記録するのだ。この点については，実際のプログラムを見てもらったほうが早い……と思ったのだが，アセンブリ言語だとさすがにわかりにくいかもしれない。というわけで，リスト3にへらへらとCで書いたKMP法による文字列照合ルーチンを用意した。関数kmp_compがテーブルの作成，kmp_execが実際の照合処理を担当する。動作試験用のメインルーチンも一緒になっているので，シンボルMAINを定義してコンパイルすれば，そのまま実行ファイルになる。出来上がるKMP.Xは，

　　KMP パターン

の形式で起動するとテキストの入力待ちになるから，適当な文字列を入力する。その文字列中にパターンと一致する部分があれば(最初のひとつが)青字で表示される。

テーブル作成部の動作はKMP法そのものだから，先に照合部である関数kmp_execを見てもらおう。変数ｉがテキスト側，ｊがパターン側の注目位置を表す。Cでは文字列を配列としてでも，ポインタを使ってでも扱えるわけだが，ここではわかりやすく配列に統一してある。ｉ，ｊはそれぞれテキスト，

リスト4　KMP.S

```
  1: *          KMP法による文字列照合
  2:
  3:            .xdef   kmp_comp
  4:            .xdef   kmp_exec
  5: *
  6:            .offset 0
  7: *
  8: PATLEN:    .ds.w   1                       *パターン長
  9: TABLE:     .ds.b   256                     *再比較開始位置のテーブル
 10: PAT:       .ds.b   256                     *パターン
 11: SIZEofWORK:
 12:
 13: *
 14: *          前処理
 15: *
 16:            .offset 4
 17: pat:       .ds.l   1                       *照合パターン
 18: *
 19:            .text
 20:            .even
 21: *
 22: kmp_comp:
 23: SAVREGS =          d1-d2/a0-a2
 24: SAVSIZ  =          (2+3)*4
 25:            movem.l SAVREGS,-(sp)
 26:
 27:            movea.l pat+SAVSIZ(sp),a2
 28:                                            *a2 = パターン
 29:            lea.l   work,a0                 *a0 = ワーク先頭
 30:            lea.l   PAT(a0),a1              *a1 = パターン格納領域
 31:            moveq.l #0,d0                   *d0.lの上位ワードをクリア
 32:            moveq.l #0,d1                   *d1.lの上位ワードをクリア
 33:
 34:            move.w  #256-1,d1               *パターンを
 35:            move.w  d1,d0                   *  ワークにコピー
 36: cpylp:     move.b  (a2)+,(a1)+             *
 37:            dbeq    d1,cpylp                *
 38:            bne     toolng                  *長すぎる
 39:
 40:            lea.l   PAT(a0),a2              *a2 = パターン
 41:            sub.w   d1,d0                   *d0 = d1 = パターン長
 42:            move.w  d0,d1                   *
 43:            move.w  d1,(a0)+                *パターン長を記憶
 44:            beq     cretn                   *パターンが空文字列だった
 45:                                            *a0 = テーブル先頭
 46:
 47:                               *テーブル作成
 48:            moveq.l #1,d0                   *d0 = 被照合側注目位置
 49:            moveq.l #0,d1                   *d1 = パターン注目位置
 50:            move.w  d1,(a0)                 *table[0] = table[1] = 0
 51: cloop:     move.b  0(a2,d0),d2             *
 52:            beq     cbreak                  *
 53:            cmp.b   0(a2,d1),d2             *
 54:            bne     comp2                   *
 55:                               *一致中
 56:            addq.b  #1,d1                   *d1文字一致している
 57: comp0:     addq.b  #1,d0                   *
 58: comp1:     move.b  d1,0(a0,d0)             *再比較開始位置をテーブルに登録
 59:            bra     cloop
 60:                               *不一致が見つかった
 61: comp2:     move.b  d1,d2                   *パターン先頭での不一致？
 62:            beq     comp0                   *そうなら再比較開始位置は先頭
 63:                               *パターン途中での不一致
 64:            move.b  0(a0,d1),d1             *テーブルから再比較開始位置を得る
 65:            bra     cloop
 66:
 67: cbreak:    movea.l a0,a1                   *絶対的な再比較開始位置から
 68:            move.w  -(a0),d1                *  相対的なポインタ移動量へ
 69:            subq.w  #1,d1                   *  変換しておく
 70:            bcs     cdone                   *
 71:                                            *
 72:            moveq.l #0,d2                   *
 73: sloop:     move.b  (a1),d0                 *
 74:            beq     snext                   *←先頭まで戻る場合を特別扱い
 75:            move.b  d2,d0                   *
 76:            sub.b   (a1),d0                 *d0 = ポインタ移動量
 77:            addq.b  #1,d0                   *←あとで(areg)++する都合
 78: snext:     move.b  d0,(a1)+                *
 79:            addq.b  #1,d2                   *
 80:            dbra    d1,sloop                *
 81:
 82: cdone:     move.w  (a0),d0                 *d0.l = パターン長
 83: cretn:     movem.l (sp)+,SAVREGS
 84:            rts
 85: *
 86: toolng:    moveq.l #-1,d0                  *エラーを返す
 87:            bra     cretn                   *
 88:
 89: *
 90: *          照合処理本体
 91: *
 92:            .offset 4
 93: txt:       .ds.l   1                       *テキスト
 94: *
 95:            .text
 96:            .even
 97: *
 98: kmp_exec:
 99: SAVREGS =          d0-d3/a1-a2
100: SAVSIZ  =          (4+2)*4
101:            movem.l SAVREGS,-(sp)
102:
103:            movea.l txt+SAVSIZ(sp),a0
104:                                            *a0 = テキスト先頭
105:            lea.l   work,a1                 *a1 = ワーク先頭
106:            move.w  (a1),d1                 *d1 = パターン長
107:            beq     qretn                   *パターン長が0だった
108:            lea.l   PAT(a1),a1              *a1 = パターン先頭
109:            move.b  (a1)+,d3                *d3 = パターン先頭文字
110:            moveq.l #0,d2                   *d2.wの上位バイトをクリア
111:
112: loop0:     move.b  (a0)+,d0                *終端に達するか
113:            beq     nmatch                  *  パターン先頭の文字と
114: retry0:    cmp.b   d3,d0                   *  一致する文字が見つかるまで
115:            bne     loop0                   *  ポインタを進める
116:
117:            movea.l a1,a2                   *a2 = パターン第2文字目
118: loop1:     move.b  (a2)+,d2                *パターンと順に比較照合する
119:            beq     match                   *全文字一致した
120:            move.b  (a0)+,d0                *
121:            beq     nmatch                  *途中でテキストが尽きた
122: retry1:    cmp.b   d2,d0                   *
123:            beq     loop1                   *
124:
125:                               *不一致が検出された
126:            move.b  -256-1(a2),d2           *d2 = ポインタ戻し量
127:            beq     retry0                  *0ならパターン先頭まで戻る
128:            suba.w  d2,a2                   *0でなければその分戻る
129:            move.b  (a2)+,d2                *
130:            bra     retry1
131:
132:                               *照合失敗
133: nmatch:    moveq.l #-1,d0                  *N = 1
134:            bra     retn
135:
136:                               *照合成功
137: match:     suba.w  d1,a0                   *a0 = 照合成功位置
138: qretn:     moveq.l #0,d0                   *N = 0
139:
140: retn:      movem.l (sp)+,SAVREGS
141:            rts
142: *
143:            .bss
144:            .even
145: *
146: work:      .ds.b   SIZEofWORK
147:
148:            .end
```

パターンの何文字目(ただし0から数える)に注目しているかを保持する。39行でi，jを0にして注目位置をリセットし，テキストとパターンの頭を揃えたら，テキスト側の注目位置が文字列末に達するまでのあいだループする。

ループの中では，まず，パターン末尾まで比較が済んだかどうかを調べる(40行)。もし，末尾に達していたら照合成功を意味するので，その位置を返す(41行)。テキスト側の注目位置はパターンの長さ分だけ進んでいるため，その分を補正していることに注意したい。

まだパターンが残っていたら，テキストとのあいだで1文字比較する(42行)。一致する場合は，両者の注目位置を進める(43〜44行)。一致しなかった場合は，テーブルを参照して，再比較を開始するパターン上の位置を取り出す(48行)。例外的に，パターン先頭での不一致だった場合はテキスト側の注目位置を進めなければならないので，そうする(45〜46行)。

ここまでの話が飲み込めれば，テーブル作成部の動作も見えてくるだろう。21〜28行のループと39〜49行のループの中身は基本的には同じ形をしている。異なるのは，パターン同士を照合していることと，一致しているあいだ，どこまで一致したかをどんどんテーブルに登録していること，この2点だ。パターンを1文字ずらして重ねたところ(i＝1，j＝0)から照合を始め，ループを抜けたときには，パターンの何文字目で不一致が見つかったら再比較を何文字目から行うか，のテーブルが出来上がっている。まだ納得できない人は紙と鉛筆を持って，動作を追ってみてほしい。

では，アセンブリ言語版をリスト4に示す。実装上の理由で，パターンの最大長は255バイト(＋終端を表す1バイトの00_H)に制限した。それ以上長い場合，サブルーチンkmp_compはエラーで戻る。これは，再比較開始位置のテーブルをバイト配列とした都合だ。ワード配列，なんならロングワード配列にすればこの制限を緩めることはできるのだが，そうすると68000ではテーブル参照時にインデックスを2倍(あるいは4倍)する処理が必要になり，バイト配列に対するアクセスより遅くなる。また，再比較開始位置のテーブルはパターンの最大長分の要素数が必要だから，より長いパターンを許すとなるとメモリもそれだけ用意しなければならない。というわけで妥協して，どうせならというのでテーブルがバイト配列であることを積極的に利用し，最適化してみた。

図4　KMP法のパターン移動量の最適化

```
パターン＝ABBABBの場合

最適化なし          最適化あり

A ?                 A B̅
A                   A

A B ? ?             A B B̅ ?
  A B                 A B

A B B ? ? ?         A B B A̅ ? ?
    A B B               A B B

A B B A ? ? ?       A B B A B̅ ? ? ?
    A B B A             A B B A

A B B A B ? ? ?     A B B A B B̅ ? ? ? ?
      A B B A B           A B B A B
```

テーブル作成部はC版をほとんどそのままハンドコンパイルした格好をしている。多少，効率改善の余地はあるだろうが，今回はこのまま押し通す。48〜65行がC版の21〜28行に対応し，array [index] のような配列参照は「0(an, dn)」のようなインデックスつきアドレスレジスタ間接形式で置き換えられている。

67行以下にC版にはないコードがあるが，ここでは照合部本体の処理を高速化するためにテーブルを少々加工している。文字列は配列として添え字でアクセスするよりもポインタで直に操作したほうが効率がよいので，テーブルの内容もそれに合わせているのだ。具体的には，再比較を開始する位置の添え字を相対的なポインタの移動量(＝注目点をどれだけ戻すか)へと変換している。77行で1を足しているのは，あとの処理でポストインクリメントする分の補正だ。あと，この部分にはもうひとつ小細工が施されている。再比較開始位置がパターン先頭(＝0)だった場合は，相対的なポインタ移動量へは変換せずに0という値をそのまま残すようにしてあるのがそれだ(74行)。こうした理由はあとで明らかになる。

照合部では効率を優先し，C版に対して，文字列のアクセス方法からループ構造まで手を入れてある。前述のように文字列は添え字でではなく，ポインタで直にアクセスするように変更した。ここで問題になるのは，不一致が検出されたときのテーブル参照だ。テーブルは不一致が起きた位置をインデックスとしてアクセスするので，ふつうにやると，注目点を指すポインタからパターン先頭アドレスを引いていったんテーブルのインデックスを得て，それにテーブル先頭アドレスを加える，という数段階の手間を必要とする。でなければ，パターンを指すポインタとは別にテーブル参照用のポインタ(あるいはインデックス)を用意して，両者をつねに連動させるか，だ。後者の場合，テーブル参照自体は高速に行えるが，パターンを指すポインタとテーブルを指すポインタの連動にちまちまと時間を食う。この点，リスト4ではテーブルもパターン文字列もバイト配

リスト5　KMP2.S(リスト4からの変更点のみ)

```
57: comp0:  addq.b  #1,d0              *
 *          move.b  0(a2,d1),d2        *次の文字も一致するなら
 +          cmp.b   0(a2,d0),d2        *
 *          bne     compl              *
 *          move.b  0(a0,d1),0(a0,d0)  *その再比較開始位置を流用
 +          bra     cloop
58: compl:  move.b  d1,0(a0,d0)        *再比較開始位置をテーブルに登録
```

列であることを利用して，シンプルにまとめている。6～11行に示したように，テーブルとパターン(のコピー)の格納用には256バイトずつのメモリが用意されており，両者はこの順序で並んでいる。したがって，パターンの途中を指すポインタから256を引けば，対応するテーブル上の位置がすぐに求められる。実際には，リスト4ではポインタをポストインクリメントしている関係で，もう1バイト引いたメモリを参照している(126行)。

ループ構造に関しては，リスト1，2同様にパターン先頭1文字を特別扱いし，その1文字を探してから残りの比較をするという2段構成にした。ただ，KMP法では，2段目のループ中でもパターン側のポインタが先頭に戻る場合があり，単純にループを2段構成にしただけでは意味がない。1段目のループをいったん抜けると，2段目の遅いループだけで処理が進んでしまう。パターン側のポインタを先頭に戻すことになったら，すかさず2段目のループを抜けて，先頭1文字を探すループに舞い戻るような細工が必要だ。ここで，テーブル作成時にパターン先頭へ戻る場合を特別扱いした効果が現れる。kmp_compで作成したテーブルでは0という特別な値によりパターン先頭へ戻ることを表すようにしたことで，判定が簡単になっているのだった(127行)。

さて，リスト3，4のkmp_compが作成するテーブルには微妙な無駄があり，これを最適化すれば，多少効率は上がる。図4を見てほしい。左側が最適化しないふつうの場合だ。下から2段目は，もし5文字目の“B”との比較で不一致が見つかったらパターンを3文字分ずらすことを意味する。しかし，ずらした結果，いま“B”と不一致を起こした文字“？”と重なるのはまたしても“B”だ。この比較は当然失敗し，もう一度テーブルを引いてパターンを移動しなければならなくなる。同様の無駄はいちばん下の段でも見られる。このような無駄を省くには，最初から一歩先を読んで，図4右側のように移動してしまえばよい[3]。この最適化は，テーブルを作成する過程で再比較位置を登録する際に，その再比較位置の文字とつぎに比較される文字を比べて，同じだったら「再比較位置で不一致が見つかったときの再比較位置」を流用することで実現される。アセンブリ言語版では，57～58の2行をリスト5のように変更すればよい。

ここで読者は，一段だけではなくその先，そのまた先まで考慮したほうがよいのではないか，と考えるかもしれない。しかし，テーブルは先頭側から出来上がっていくので，流用した再比較位置はすでに最適化されている。したがって，1ステップ先を読めばもう十分だ。

なお，図4右側を見ると，4文字目の“A”で不一致が見つかった場合だけは，まだ“A”と重ねる無駄をやっている。ここも最適化できればよいのだが，実現は難しい。パターン先頭との不一致になるのでテキスト側の注目位置を進めなければならず，統一的な形にはまとまりそうもない。どうしてもというのであれば，テーブルに「テキストの注目位置を進めてからパターン先頭と比較せよ」という意味を表す，特別な値を導入することになるだろうが，それでは比較の手間が増える。いま問題にしているような無駄は，パターンの先頭と末尾の文字が同じ場合にしか起きないから，無理をしても平均の性能は向上せず，むしろ遅くなってしまうだろう。ここは，無視するのが正解のようだ。

KMP法の性能に関しては，理論上，照合時の文字比較回数が最悪でも2×n回以下であることが知られている。テーブル作成時に行われる最大2×m回[4]の比較と合わせても，2(m＋n)回以下だ。これは単純法のm×n回に比べて大きな進歩といえる。しかし，残念ながら，現実にはKMP法はあまり速くない。というのも，KMP法でパターンを大きく移動できるのは，パターン先頭が何文字か一致したあとで不一致が見つかった場合に限られるからだ。一般に部分一致は起こりにくいので，KMP法が効果的に働くことはほとんどない。アルゴリズムが複雑で，テーブル作成にも余分な時間がかかる分，へたなプログラムでは単純法よりずっと遅くなることもある。リスト4，5はKMP法の実装例としてはかなり高速に仕上がっているはずなのでそれほどひどくはないが，それでも“とんとん”がいいところだ。

とはいっても，KMP法の最悪の場合の性能のよさには注目するだけの価値がある。また，ある種の応用ではKMP法がそれなりに実用的なこともある。たとえば，2種類の文字しか使わない2進文字列を扱う場合には，高い確率で部分一致が起きるので，KMP法が効果的に働くだろう。さらに，KMP法にはテキストを走査するポインタが逆戻りすることがないという特徴がある。この特徴により，KMP法はメモリに納まらないような長いテキストを扱うのに適しているといわれる。事実，Morrisはテキストエディタを作成中に，ポインタが逆戻りしないようなアルゴリズムを考えていてKMP法にたどり着いたという。

ボイヤー-ムーア法

BoyerとMoore，および，それとは独立にGosperが考案したBoyer-Moore(BM)法[5]は，現時点で，実用上，最も速い文字列照合アルゴリズムだ。このアルゴリズムを使っていない文字列検索ツールやエディタは，もぐり，といってもいいすぎではない。今月，このアルゴリズムだけはぜひ自分のものにしてもらいたいと思う。

BM法の最大の特徴は，パターンを末尾側から逆方向に比較していくことにある。このコロンブスの卵的な発想により，BM法では比較回数を劇的に減らすことに成功している。

最初にテキストとパターンの先頭を揃えるところまではほかのアルゴリズムと変わらない。ふつうな

3）図中“B̄”のような上線つきの文字は，「～以外の文字」を表すとする。

4）テーブルを最適化する版では3m回。

5）Boyer-Moore-Gosper法と呼ばれることもないではない。

ら，ここからパターン先頭とテキスト先頭を比較するわけだが，BM法ではパターン末尾，(先頭からm文字目）の文字と，テキストのm文字目の文字を比較する。一致するようなら注目点を戻しつつテキストとパターンを末尾側から逆方向に比較していく。不一致が検出されたら，その不一致を引き起こしたテキスト中の文字に注目する。もし，その文字がパターン中に含まれるのなら，両者が重なる位置にまでパターンを右にずらせる。そうしないと，ふたたび同じ位置で不一致になるのが目に見えているからだ。また，同様の理由により，不一致を引き起こした文字がパターン中に含まれない場合は，その文字とパターンが重ならないところまで一気にパターンを右にずらすことができる。ずらしたら，ふたたびパターン末尾から比較していく。この様子を図5に示した。

図5では，最初のステップでパターン末尾の文字“U”とテキスト中の文字“Y”を比較すると，すぐに不一致が検出される。“Y”はパターン中にあるから，“Y”どうしが重なる位置までパターンを3文字分ずらす。すると，ふたたび“U”と“Y”を比べることになるので，また3文字分ずらす。つぎには“U”と“A”を比べて不一致が検出される。“A”はパターン中には含まれないので，“A”と重ならない位置までパターンを5文字分ずらす。図5の例ではこの時点でパターンとテキスト中の部分文字列が一致し，照合が成功する。

KMP法同様，BM法でもパターンの移動量はあらかじめテーブルにしておく。KMP法のテーブルは不一致を起こしたパターン上の位置をインデックスとして参照したが，BM法では不一致を起こしたテキスト側の文字がインデックスとなる。このテーブルには，各文字がパターンの右から何文字目に現れるか(右端の文字位置を0とする)を登録しておく。パターン中に同じ文字が複数ある場合は，その最も右側の位置を採用する。パターンに含まれない文字については，パターンの長さをそのまま登録する。図5の例の場合だと，パターン長は5で，

```
43210
KYOKU
```

だから，

```
K         1
Y         3
O         2
U         0
それ以外  5
```

となる。

この値はそのままパターンの移動量として使える値ではないことに注意したい。図5の例ではテーブルの値と移動量が等しいので錯覚してしまうかもしれないが，これはたまたまパターン末尾でだけ不一致を検出しているからそう見えるだけだ。実際には，パターン末尾から2文字目で不一致が検出されたら，テーブルの値から1を引いた値を移動量とし，3文字目だったら2を引く，というような補正が必要になる。これはテーブルに登録した値の意味を考えてみれば自明だろう。テーブルにはパターン末尾からの距離が登録されているのだから，末尾以外からなら距離は縮まるのだ。別のいい方をすると，BM法のテーブルは，“不一致を起こした位置”を基準にして，そこから右に何文字目にパターンの“末尾”がくるよう移動するか，を保持している。

テーブル作成の手順は非常に簡単だ。まず，文字の種類分の大きさの配列を用意し，これをパターンの長さで埋めて初期化しておく。それから，パターンの長さ－1を初期値とするカウンタを設けて，パターンを先頭から見ていき，見つけた文字に対応するテーブル上の位置にその時点でのカウンタの値を登録してはカウンタから1減じる。パターン中に同一の文字が複数存在する場合，同じ文字が出てくるたびにテーブルを上書きすることになるので，テーブルには最も右側の位置が残る。

さて，ここまでの話にはまだ穴が2つほど残っている。ひとつはアルゴリズム上の穴だ。図6を見てほしい。図6ではパターン末尾から3文字目の“O”と“K”の比較で不一致が検出される。“K”はパターンの右から2文字目にあるので末尾からの距離は1。そこで現在の注目点＋1にパターンの末尾を合わせると，パターンは左に逆戻りする。パターン中，比較位置よりも右に不一致を起こした文字が含まれているとこのようなバックスライドが起きる。バックスライドをそのまま採用してしまうと堂々巡りに陥るから，どんな場合でもパターンは右に最低1文

図5　BM法による文字列照合

```
1) TOKKYOKYOKAKYOKU  ←テキスト
   KYOK[U]           ←パターン

2) TOKKYOKYOKAKYOKU
      KYOK[U]

3) TOKKYOKYOKAKYOKU
         KYOK[U]

4) TOKKYOKYOKAKYOKU
           KYOKU
```

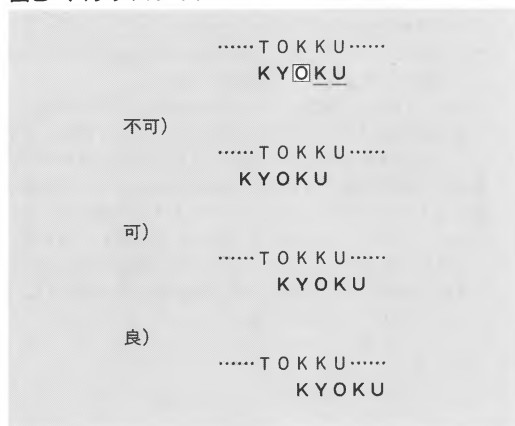



図6　バックスライド

字分は移動しなければならない。で，ここが微妙なところなのだが，バックスライドが起きたときには，パターンを右に1文字だけではなく“2文字分”ずらしてしまってかまわないことがわかっている。実用上は，その事実だけを知っていればよいが，それでは納得できないという人のためにコラムで理屈をこねておいた。

ちょっと補足しておくと，実はここまでで説明したのはBM法の半分だけを利用した簡略版だ。Boyer-Mooreのオリジナルの方法ではテーブルをもうひとつ用意する。その第2のテーブルには，KMP法に似た考え方で，一致した部分文字列を基にしたパターン移動量を登録しておき，2つのテーブルのうち，どちらか移動量の大きいほうを使うようになっている。この完全なBM法ではバックスライドは起きない。ただ，KMP法同様，第2のテーブルはあまり平均の性能には貢献せず，そのくせに作成が面倒だったりするので，通常，これまで説明したような簡略版が利用される。

BM法のもうひとつの穴は実装時に露見する。BM法ではテキストを1文字ごとに調べるわけではないので，文字列の終端コードに頼ったのでは，テキストの末尾を越えて走査が進んでしまう可能性がある。このことが，実用上，大きな障害となる。これのどこが問題なのかと不思議に思うかもしれない。文字列照合サブルーチンの先頭でテキストの文字数を数えておけば何の不都合もないように見える。そのうえで，テキストの末尾にパターン末尾の文字をいくつか並べて番人を置くようにすれば，効率の低下も抑えられるだろう。

しかし，終端コードで末尾を表す形式の文字列の長さを数えるのは，とてもコストのかかる処理だ。先頭から終端コードに出会うまで文字列を走査するわけであり，文字列の長さに応じた回数のメモリアクセスを必要とする。しかも，テキストの先頭付近ですぐに照合が得られる場合には，せっかくテキストを全部走査して文字数を数えても，その時間のほとんどは無駄になる。さらに照合成功位置の直後から再検索したりすると，再度末尾まで走査することになり，無駄を積み重ねる結果となる。

解決策は，サブルーチン側でテキストの長さを数えるのをやめて，メインルーチン側が引数として，テキストの長さなり末尾のアドレスなりを渡すことだ。結局はただ負担をメインルーチン側に押しつけた格好だが，通常，メインルーチンは扱うデータについてサブルーチン側よりも多くの情報を持っている。文字列長ぐらい，ほかの処理の過程で副作用として得られることも多い。それを覚えておけば，わざわざ数え直す必要はないわけだ。たとえば，テキストエディタなら，各行の長さという情報をつねに保持しているだろうから，これがそのまま流用できる。また，ファイルからの文字列検索ツールでは，ファイルをある程度の単位でバッファにまとめて読み込み，そこから行を切り出すのが自然だが，その行切り出しの過程で文字数がわかる。逆にいうと，わざわざテキストの長さを数えなければならない状況では，BM法を使うメリットはない。

というあたりで，BM法による文字列照合ルーチンの実例を示そう。縁起ものなので，まずC版を用意した(リスト6)。ぱっと見て，アルゴリズムの流れを確認しておいてほしい。で，リスト7がアセンブリ言語版。今月の最終版ということで，きちんと

バックスライド時の対応

簡略BM法でバックスライドが起きたときには，2文字パターンをずらしてよいことを簡単に示す。

あと1文字分パターンを右にずらせば照合が得られるという状況を想定する。

```
    ……XABCDEFG……
       ABCDEFG
```

ここで，AやBは文字“A”,“B”そのものではなく，何かしらの1文字を表す記号と考えてほしい。同じ記号は同じ文字を表す。

この状態ではバックスライドが起きないことを示すことができれば，もしバックスライドが起きたら2文字分パターンを動かしてよいことが証明される。

まず，パターン末尾での不一致が発生した場合には，どんな場合でもバックスライドは起きないことを確認しておく。バックスライドは不一致を引き起こした文字が注目位置よりも右に存在する場合にのみ起こる。パターン末尾の右にはもう文字がないから，バックスライドは起こりようがない。そこで，このケースは最初から除外して考える。

パターン末尾側から比較していったら適当なところ，たとえば，後ろから4文字目で不一致が見つかったとしよう。

```
    ……XABCDEFG……
       ABCDEFG
          ↑
         注目点
```

後ろ3文字は一致したのだから，

F＝G
E＝F
D＝E

ということは，

D＝E＝F＝G

だ。そこで，これらの文字をすべてDに統一しておく。

```
    ……XABCDDDD……
       ABCDDDD
          ↑
         注目点
```

で，CとDとの比較で不一致が見つかり，バックスライドが起きた，と仮定する。バックスライドが起きるためには，不一致を起こした文字Cが，パターン中のすでに一致が確認された部分，この例では末尾の3文字に含まれていたことを意味する。末尾の3文字はすべて同じ文字Dだから，

C＝D

となる。ところが，これではCとDが不一致を起こすはずがなく，仮定と矛盾する。ゆえに，この場面ではバックスライドは起こりえない。したがって，バックスライドが起きた場合に照合が得られる可能性のある位置は，少なくとも2文字分右にパターンをずらした位置以降にある。

日本語にも対応している。

サブルーチンbm_compがテーブル作成部，bm_execが実際の照合部本体だ。bm_compについては，これまでの説明，および，C版と同様なので，特に解説するまでもない。46〜48行でテーブルをパターン長で初期化したあと，54〜58行でテーブルを作り上げる。ちょっと工夫したのが60〜64行で，ここではパターンの先頭と末尾を反転したコピーを作っている。たとえば，パターンが，

　　a b c d e

なら，

　　e d c b a

のような文字列を作る。これは，ポストインクリメントつきのアドレスレジスタ間接形式を有効に活かすための処置だ。BM法では比較をパターン末尾側から行うから，ふつうならプリデクリメントつきのアドレスレジスタ間接形式を利用することになるが，たびたび触れるように，68000ではプリデクリメントするよりも，ポストインクリメントしたほうが多少速い。そこで，何度も参照されるパターン側の走査時にポストインクリメントできるよう細工しているのだ。なお，情けないことにいま気づいたのだが，リスト7にはパターンのコピーを作ってからそれを反転するという無駄が見られる。大勢には影響しないとはいえ，引数で渡されたパターン文字列から，直接，反転パターンを作ったほうが効率はよいだろう。すかさず，修正しておいてほしい。

照合部のbm_execは，例によって，アルゴリズム上の鍵となる位置であるパターン末尾の1文字を特別扱いすることで，効率を稼いでいる。106〜111行のループでパターン末尾の文字を探し，114〜121行で残りを比較するわけだ。106〜111行のループではパターンそのものにはアクセスしていないことに注意してほしい。ここでは，テキストの注目位置を取り出して，その文字をインデックスとしてテーブルを参照し，その分注目位置をずらす，という処理しかしていない。パターン末尾の文字かどうかは，テーブルから引いた値が0かどうかで判別している。また，このループではパターン末尾のみを扱うので，バックスライドは考慮しなくてよくなっている。

残りの比較をする119〜121行のループは見てのとおりだ。先に触れたように，パターンは反転してあるので，ポストインクリメントつきのアドレスレジスタ間接形式で走査している。

全文字一致したら，138行にくる。ここで照合が得られた位置が2バイト文字の途中から始まっていないかどうかをテストする。考え方は単純だ。照合が得られた先頭の位置から，テキスト先頭方向に遡り，“シフトJIS漢字コードの第1バイトにはならない文字”を探す。見つけたその文字は，2バイト文字の第2バイトか，1バイト文字のどちらかだ。どちらにしろ，文字はこの位置で完結しており，直後からつぎの文字が始まっている。で，シフトJIS漢字コードの第1バイトをスキップしたわけだから，その文字と照合が得られた位置のあいだには，あるとしたら2バイト文字だけが並んでいるはずだ。もし，それ以外のものがあるとしたら，それは生き別れになった2バイト文字の断片にほかならない。したがって，そのあいだに何バイトあるかで，照合が得られた位置が2バイト文字の途中かどうかが判断できる。偶数だったらつじつまがあっているからゴミはなく，奇数だったら尻切れになった2バイト文字があるわけだ。

この方法は，テキストの正当性，つまり，文字としては正しくない変なデータが存在しないことを仮定しているので絶対の方法ではないが，ふつうのテキストならこの条件はつねに満たされると考えてよいだろう。あとは，ある1バイトデータが，シフトJIS漢字コードの第1バイトかどうか，なるべく高速にテストできればよい。

このテストは文字コードの比較で簡単に実現でき

リスト6　BM.C

```
 1: /*  BM法による文字列照合     */
 2:
 3: #define NULL    ((void *)0)
 4: #include    <string.h>
 5: #include    <limits.h>
 6:
 7: #define MAXLEN  256
 8:
 9: static int bm_len;
10: static unsigned char bm_pat[MAXLEN];
11: static unsigned char bm_table[UCHAR_MAX + 1];
12:
13: int bm_comp(unsigned char *pat)
14: {
15:     int i, j;
16:
17:     if ((bm_len = strlen(pat)) >= MAXLEN)
18:         return -1;
19:
20:     strcpy(bm_pat, pat);
21:
22:     for (i = 0; i <= UCHAR_MAX; i++)
23:         bm_table[i] = bm_len;
24:
25:     for (i = 0, j = bm_len - 1; i < bm_len; i++, j--)
26:         bm_table[pat[i]] = j;
27:
28:     return bm_len;
29: }
30:
31: unsigned char *bm_exec(unsigned char *head, unsigned char *tail)
32: {
33:     unsigned char *p, *lastp;
34:     int j;
35:
36:     if ((bm_len) == 0)
37:         return head;
38:
39:     for (p = head + bm_len; p <= tail;) {
40:         for (j = bm_len, lastp = p; *--p == bm_pat[--j];)
41:             if (j == 0)
42:                 return p;
43:
44:         p += bm_table[*p] + 1;
45:         if (p <= lastp)
46:             p = lastp + 2;
47:     }
48:     return NULL;
49: }
50:
51: #ifdef  MAIN
52:
53: #include    <stdio.h>
54: #include    <stdlib.h>
55:
56: void main(int argc, unsigned char **argv)
57: {
58:     int n;
59:     unsigned char buff[256], *p;
60:
61:     if (argc < 2)
62:         exit(EXIT_FAILURE);
63:
64:     n = bm_comp(*++argv);
65:     while (fgets(buff, 256, stdin) != NULL)
66:         if ((p = bm_exec(buff, strchr(buff, '¥0'))) != NULL)
67:             printf("%.*s¥x1b[31m%s¥x1b[m%s",
68:                 p - buff, buff, *argv, p + n);
69:
70:     exit(EXIT_SUCCESS);
71:
72:
73: #endif
```

るが，シフトJIS漢字コードの第1バイトは途中に隙間があるので，比較を何回か行わなければならず，あまり効率はよさそうにない。この場面では，文字種の判別テーブルを使うのが常道だ。256バイトのテーブルを用意しておき，文字コードに応じた位置を引くとすぐに文字の種類がわかるようにしておく。いまの目的では，単純に，シフトJIS漢字コードの第1バイトだったら1，そうでなければ0といった具合にテーブルを作っておけばよい。実際には，それだけの目的で256バイトのメモリを使うのはもったいないので，たとえば，第0ビットが1だったら半角数字，第1ビットが1だったら半角英大文字，というようにビットごとに意味をもたせるのがよいだろう。こうして作ったテーブルはいろいろな場面で流用できる。リスト7では，このようなテーブルが別に用意されているものとし，テーブルから引いた値の第7ビットが1かどうかで，シフトJIS漢字コードかどうかを判定するようにした。第7ビットは符

リスト7　BM.S

```
  1: *          BM法による文字列照合
  2:
  3:            .xdef   bm_comp
  4:            .xdef   bm_exec
  5:            .xref   ctypetable
  6: *
  7:            .offset 0
  8: *
  9: PATLEN:    .ds.w   1                       *パターン長-1
 10: TABLE:     .ds.b   256                     *移動量のテーブル
 11: PAT:       .ds.b   256                     *パターン（反転）
 12: SIZEofWORK:
 13:
 14: *
 15: *          前処理
 16: *
 17:            .offset 4
 18: pat:       .ds.l   1                       *パターン
 19: *
 20:            .text
 21:            .even
 22: *
 23: bm_comp:
 24: SAVREGS =          d1/a0-a2
 25: SAVSIZ  =          (1+3)*4
 26:            movem.l SAVREGS,-(sp)
 27:
 28:            movea.l pat+SAVSIZ(sp),a1
 29:                                        *a1 = パターン
 30:            lea.l   work,a0             *a0 = ワーク先頭
 31:            lea.l   PAT(a0),a2          *
 32:            moveq.l #0,d0               *d0.lの上位ワードをクリア
 33:
 34:            move.w  #256-1,d1           *パターンを
 35:            move.w  d1,d0               *  ワークにコピー
 36: cpylp:     move.b  (a1)+,(a2)+         *
 37:            dbeq    d1,cpylp            *
 38:            bne     toolng              *長すぎる
 39:
 40:            sub.w   d1,d0               *d0 = パターン長
 41:            move.w  d0,d1               *
 42:            subq.w  #1,d1               *d1 = パターン長-1
 43:            move.w  d1,(a0)+            *覚えておく
 44:            bmi     cretn               *パターンが空文字列だった
 45:
 46:            move.w  #256-1,d1           *テーブルを
 47: filllp:    move.b  d0,(a0)+            *  パターン長で埋める
 48:            dbra    d1,filllp           *
 49:
 50:            movea.l a0,a1               *a1 = a2 = パターン
 51:            movea.l a1,a2               *
 52:            lea.l   -256(a0),a0         *a0 = テーブル先頭
 53:
 54:            moveq.l #0,d1               *テーブル作成
 55:            subq.w  #1,d0               *
 56: complp:    move.b  (a2)+,d1            *
 57:            move.b  d0,0(a0,d1)         *
 58:            dbra    d0,complp           *
 59:
 60: revlp:     move.b  -(a2),d1            *あとの処理に備えて
 61:            move.b  (a1),(a2)           *  パターンを反転しておく
 62:            move.b  d1,(a1)+            *
 63:            cmpa.l  a2,a1               *
 64:            bcs     revlp               *
 65:
 66:            move.w  -(a0),d0            *
 67:            addq.w  #1,d0               *d0.l = パターン長
 68:
 69: cretn:     movem.l (sp)+,SAVREGS
 70:            rts
 71: *
 72: toolng:    moveq.l #-1,d0              *エラーを返す
 73:            bra     cretn               *
 74:
 75: *
 76: *          照合処理本体
 77: *
 78:            .offset 4
 79: txt:       .ds.l   1                   *テキスト
 80: txtend:    .ds.l   1                   *テキスト末尾
 81: *
 82:            .text
 83:            .even
 84: *
 85: bm_exec:
 86: SAVREGS =          d0-d3/a1-a6
 87: SAVSIZ  =          (1+6)*4
 88:            movem.l SAVREGS,-(sp)
 89:
 90:            movem.l txt+SAVSIZ(sp),a0/a5
 91:                                        *a0 = テキスト先頭
 92:                                        *a5 = テキスト末尾
 93:            lea.l   work,a1             *a1 = ワーク先頭
 94:            lea.l   PAT+1(a1),a2        *a2 = パターン末尾の1文字手前
 95:            move.w  (a1)+,d1            *d1 = パターン長-1
 96:            bmi     qretn               *パターン長が0だった
 97:            lea.l   ctypetable,a6       *a6 = 文字種判別用テーブル
 98:
 99:            move.l  a0,d3               *d3 = テキスト先頭
100:            adda.w  d1,a0               *a0 = パターン末尾との重ね位置
101:            subq.w  #1,d1               *d1 = パターン長-1のdbraカウンタ
102:
103:                              *パターン末尾と一致する文字を探す
104:            moveq.l #0,d0               *d0.wの上位バイトをクリア
105:            bra     next0               *
106: loop0:     move.b  (a0),d0             *d0 = 入力文字
107:            move.b  0(a1,d0),d0         *d0 = 対応する移動量
108:            beq     break0              *それが0ならパターン末尾の文字
109:            adda.w  d0,a0               *ポインタを移動量の分進める
110: next0:     cmpa.l  a5,a0               *末尾を越えるまで繰り返す
111:            bcs     loop0               *
112:            bra     nmatch.             *パターンは見つからなかった
113:
114: break0:    movea.l a0,a3               *a0 = パターン末尾との一致位置
115:            movea.l a2,a4               *a4 = パターン末尾の1文字手前
116:            move.w  d1,d2               *d2 = パターン長-1のdbraカウンタ
117:            bmi     zenchk              *パターン長が1だった
118:
119: loop1:     move.b  -(a3),d0            *パターン末尾側から照合
120:            cmp.b   (a4)+,d0            *
121:            dbne    d2,loop1            *
122:            beq     zenchk              *全文字一致した
123:
124:            move.b  0(a1,d0),d0         *d0 = 不一致を起こした文字の移動量
125:            adda.w  d0,a3               *その分ポインタを進める
126:            exg.l   a3,a0
127:            cmpa.l  a0,a3               *後戻りした？
128:            bcs     next0               *
129:            lea.l   2(a3),a0            *  そうなら2バイト進める
130:            cmpa.l  a5,a0               *末尾を越えたるまで繰り返す
131:            bcs     loop0               *
132:
133:                              *照合失敗
134: nmatch:    moveq.l #-1,d0              *N = 1
135:            bra     retn
136:
137:                              *照合成功位置が全角文字の
138: zenchk:                      *第2バイトではないことを確認する
139:            movea.l a3,a4               *a4 = 照合成功位置先頭
140:            move.l  a4,d2               *d2 = テキスト先頭から
141:            sub.l   d3,d2               *  照合位置までのバイト数
142:            beq     match               *先頭なら間違いない
143:
144:            subq.l  #1,d2               *テキストを遡り
145:            swap.w  d2                  *  シフトJIS漢字コートの第1バイト
146: zloop1:    swap.w  d2                  *  にはならない文字の位置を探す
147: zloop2:    move.b  -(a4),d0            *
148:            tst.b   0(a6,d0)            *
149:            dbpl    d2,zloop2           *
150:            bpl     zchk0               *
151:            swap.w  d2                  *
152:            dbra    d2,zloop1           *
153:            bra     zchk1               *
154:
155: zchk0:     addq.l  #1,a4               *a4 = 全角文字の2バイト目ではない
156:                                        *  ことが保証された文字位置
157: zchk1:     move.l  a3,d2               *d2 = その文字位置と照合成功位置の差
158:            sub.l   a4,d2               *
159:            andi.b  #1,d2               *偶数？
160:            beq     match               *  そうなら照合成功
161:
162:                              *照合成功位置は全角文字の2バイト目だった
163:            addq.l  #1,a0               *
164:            cmpa.l  a5,a0               *
165:            bcs     loop0               *
166:            bra     nmatch
167:
168:                              *照合成功
169: match:     movea.l a3,a0               *a0 = 照合成功位置
170: qretn:     moveq.l #0,d0               *N = 0
171:
172: retn:      movem.l (sp)+,SAVREGS
173:            rts
174: *
175:            .bss
176:            .even
177: *
178: work:      .ds.b   SIZEofWORK
179:
180:            .end
```

号ビットであり，ビットテスト命令を使わずに済むのがポイントだ。テーブル自体はリスト８のX-BASICプログラムで生成した。リスト８の出力はまだ不完全なので，あとからリスト９のように外部定義をつけ加え，リスト７と一緒にリンクしてほしい。

では，BM法の性能を評価してこの項を終えよう。今回採用した簡略版では，最悪の場合，m×n回の文字比較を必要とする[6]。テキスト“ＡＡ……Ａ”からパターン“ＢＡＡ……Ａ”を探す場合がこれにあたる。逆に，つねにパターン末尾で不一致が見つかり，その不一致を起こした文字がパターン中に含まれないという幸運な場合は，m文字ずつパターンを動かしていけるわけだから，比較回数はn/m回に収まる。単純法の場合と同じような考え方をすると，BM法の平均の性能はn/mのほうに近いと予想される。パターン長が長ければ，より大きくパターンを移動していけるので，効率は上がる。ただ，パターンが長くなるとパターン中にテキストと一致する文字を多く含む確率も高くなるため，単純にパターン長に反比例するというわけにもいかないだろう。ある程度のところで頭打ちになると考えられる。

性能比較

最後に今回作成したサブルーチンの性能を検証しておこう。簡単ないくつかのテストを行った結果を表１に示す。与えられた条件下でテキスト中からパターンをすべて見つけるのにかかった時間を，いくつか(テストによっては最大1000数百個程度)のパターンで測定し，平均をとった。KMP法，BM法ではテーブル作成の時間も含まれている。「リスト１改」というのは，リスト１に本文中で示した２点の改良を加えた版を指す。また，いきなり登場したリスト10は，Wirthの本にみられる簡略BM法の別バージョンだ。詳しく触れるゆとりはないが，この変形では，「パターン末尾と比較した文字」にのみ注目してパターンを移動していく。リスト10を読むときには，テーブル作成時のループが１回少なくなっている意味をよく考えてみてほしい。

なお，ほかとの兼ね合いで，リスト７，10については全角文字のチェックを外してある。もっとも，そのチェックを入れても，実行時間にはほとんど差がないことをつけ加えておく。

最初のテストＡが，通常の用途における平均性能の目安になるだろう。このテストに関しては参考までにＣ版とXCのライブラリ関数strstr[7]の結果もつけてみた。今月作成したＣ版のKMP法，BM法の文字列照合関数は速くする工夫を何もしていないのでフェアではないのだが，とりあえずGNU Ｃで適当に最適化オプションを効かせてある[8]。添え字を極力使わずにポインタを利用するようにすれば，もう１～２割は簡単に速くなるし，まだまだ工夫の余地はあるから，Ｃプログラマは検討してみてもらいたい。もっとも，アセンブリ言語版との差を大きく詰めるのは難しいだろう。ＣからBM法による文字列照合を利用したければ，Ｃから呼び出せるようにリスト７を修正したほうがよいと思う。

テストＢはBM法の効果を強調するためにのみ行った。パターンが長いほど有利というBM法の特徴が確認できる。

テストＣ以降は特殊条件下のテストだ。まず，テ

リスト8　GENCTYPE.BAS

```
 10 char table(255)
 20 int fp,i
 30 str s,s0,s1
 40 s0 = chr$(9)+".dc.b"+chr$(9)+"%"
 50 s1 = chr$(13)+chr$(10)
 60 /*
 70 settable('0', '9',        &B11)
 80 settable('A', 'F',        &B10)
 90 settable('a', 'f',        &B10)
100 settable('A', 'Z',      &B1100)
110 settable('a', 'z',     &B10100)
120 settable(&H80,&H80, &B1000000)
130 settable(&HF0,&HFF, &B1000000)
140 settable(&H80,&H9F,&B10000000)
150 settable(&HE0,&HFF,&B10000000)
160 /*
170 fp = fopen("$","c")
180 for i = 0 to 255
190      s = right$("0000000"+bin$(table(i)),8)
200      fwrites(s0+s+s1,fp)
210 next
220 fclose(fp)
230 end
240 /*
250 func settable(st,ed,flag)
260     int i
270     for i = st to ed
280         table(i) = table(i) or flag
290     next
300 endfunc
```

リスト9　MYCTYPE.S

```
 1: *           文字種判別用テーブル
 2:
 3:             .xdef    ctypetable
 4: *
 5:             .text
 6:             .even
 7: *
 8: ctypetable:
 9:             .dc.b    %00000000
10:             .dc.b    %00000000
                      :
                      :
```

表１　文字列照合アルゴリズムの性能比較　(単位：ms)

	A	B	C	D	E	F
単純法（リスト１）	351.0	353.0	283.6	106.3	2544.5	105.0
単純法（リスト１改）	343.5	344.9	242.9	105.5	2287.8	104.8
単純法（リスト２）	344.7	346.0	233.0	105.6	1721.2	104.8
KMP法（リスト４）	347.0	348.4	209.6	105.8	358.6	105.0
KMP法（リスト５）	347.0	348.4	199.8	105.9	358.0	105.1
BM法（リスト７）	124.0	73.5	242.2	24.3	189.0	954.5
BM法（リスト10）	127.5	75.9	210.7	25.9	202.6	1777.4
XCのstrstr関数	393.1	----------	----------	----------	----------	----------
KMP法Ｃ版（リスト３）	1306.7	----------	----------	----------	----------	----------
BM法Ｃ版（リスト６）	336.9	----------	----------	----------	----------	----------

Ａ　100Ｋバイトの英文テキスト中から抽出した３～15文字長の単語を同テキストから検索
Ｂ　100Ｋバイトの英文テキスト中から抽出した10～15文字長の単語を同テキストから検索
Ｃ　32Ｋバイトのランダムな２進テキストから８文字長のランダムな２進文字列を検索
Ｄ　32Ｋバイトのランダムな200進テキストから８文字長のランダムな200進文字列を検索
Ｅ　32Ｋバイトの「a」だけが並んだテキストから「aaaaaaaaaaaaaaab」を検索
Ｆ　32Ｋバイトの「a」だけが並んだテキストから「baaaaaaaaaaaaaaa」を検索

6）完全なBM法はKMP法に似た戦略を併用するために，最悪の場合の比較回数もKMP法並みに抑えられる。

7）第１引数で指定した文字列から第２引数で指定した文字列を探してその位置を返す，つまり，ちょうど今回のテーマである文字列照合を行う関数，リスト１，２のサブルーチン名もここからとっている。

8）ちなみにXCでコンパイルすると2.5倍ほど遅い。

ストCは繰り返しの多いテキストからやはり繰り返しの多いパターンを探す場合の様子をみる。2種類の文字しか含まないテキストを乱数で作成し，同様に乱数で作成した8文字長のパターン1000個と照合した。このテストではテキストの各文字位置で1/256の確率でパターンとの照合が得られ，部分的な一致ともなれば頻繁に起きる。KMP法に花を持たせるためのテストと思ってもらってよい。同じようなテストを200種類の文字に増やして行うとテストDの結果となる。今度は照合が得られる可能性は限りなく0に近い(実際，テストDではひとつも照合は得られていない)。結局，テストDはテキストを端から端まで走査する最短時間を計測した格好だ。

残りの2つのテストは各アルゴリズムの不得手とするデータを与えてみたときの様子だ。ほぼ予想どおりの結果になっている。このような特殊な設定では，平均性能の向上を狙った最適化がことごとく裏目に出るのが面白いといえば面白い(当然といえば当然)。

＊　　＊　　＊

文字列照合には，英字の大文字/小文字の同一視とか，複数パターンの並行照合といった実用上重要な応用があるのだが，これについては読者への課題としたい。とくにBM法でこれらを実現するにはどうすればいいか，検討してみてほしい。後者については，各パターンについてBM法のテーブルを作成してから，そのテーブルを重ね合わせてもうひとつテーブルを作るのがひとつの解となる。

文字列照合については文献もいろいろあるからそちらをあたってみるのもよいだろう。いまふつうに手に入る書籍で，今回の内容のおさらい程度の目的なら，Sedgewickの『ALGORITHMS』(翻訳は『アルゴリズム』のタイトルで近代科学社より出版されている)がお勧めだ。また，Wirthの『ALGORITHMS & DATA STRUCTURES』(翻訳は同社より『アルゴリズムとデータ構造』)では，さほどページは割かれていないが，本稿よりもずっと詳しい数学的解析がある。先に触れたように，簡略BM法の別版も取り上げられている。

あと，BM法については，『アスキー』誌1987年9月号に掲載された『COMPUTER LANGUAGE』誌からの翻訳記事がかなり詳しい。少々古いが，BM法について日本語で読めるものとしては，いまだにあの記事がいちばんまとまっているように思う。簡略BM法でバックスライドが起きたときに2文字ずらせるという情報は同記事から得た。

さて，次回は今月の勢いを駆って「パターン照合」方面に走るつもりだ。が，以前すっ飛ばした「木」を先に片づけておいたほうがいいような気もしている(手順前後……)。パターン照合についてどこまでやるかでも事情は変わってくるので，もう少し検討させてほしい。

リスト10　BM2.S

```
  1: *          BM法による文字列照合(別版)
  2:
  3:            .xdef   bm_comp
  4:            .xdef   bm_exec
  5: *
  6:            .offset 0
  7: *
  8: PATLEN:    .ds.w   1                       *パターン長-1
  9: TABLE:     .ds.b   256                     *移動量のテーブル
 10: PAT:       .ds.b   256                     *パターン(反転)
 11: SIZEofWORK:
 12:
 13: *
 14: *          前処理
 15: *
 16:            .offset 4
 17: pat:       .ds.l   1                       *パターン
 18: *
 19:            .text
 20:            .even
 21: *
 22: bm_comp:
 23: SAVREGS =          d1/a0-a2
 24: SAVSIZ  =          (1+3)*4
 25:            movem.l SAVREGS,-(sp)
 26:
 27:            movea.l pat+SAVSIZ(sp),a1
 28:                                        *a1 = パターン
 29:            lea.l   work,a0                 *a0 = ワーク先頭
 30:            moveq.l #0,d0                   *d0.lの上位ワードをクリア
 31:
 32:            move.w  #256-1,d1               *パターンを
 33:            move.w  d1,d0                   *  ワークにコピー
 34:            lea.l   PAT(a0),a2              *
 35: cpylp:     move.b  (a1)+,(a2)+             *
 36:            dbeq    d1,cpylp                *
 37:            bne     toolng                  *長すぎる
 38:
 39:            sub.w   d1,d0                   *d0 = d1 = パターン長
 40:            move.w  d0,d1                   *
 41:            subq.w  #1,d1                   *あとてのdbraを考慮
 42:            move.w  d1,(a0)+                *パターン長を記憶
 43:            bmi     cretn                   *パターンが空文字列だった
 44:
 45:            move.w  #256-1,d1               *テーブルを
 46: filllp:    move.b  d0,(a0)+                *  パターン長で埋める
 47:            dbra    d1,filllp               *
 48:
 49:            movea.l a0,a1                   *a1 = a2 = パターン
 50:            movea.l a1,a2                   *
 51:            lea.l   -256(a0),a0             *a0 = テーブル先頭
 52:
 53:            moveq.l #0,d1                   *テーブル作成
 54:            bra     compnx                  *
 55: complp:    move.b  (a2)+,d1                *
 56:            move.b  d0,0(a0,d1)             *
 57: compnx:    subq.w  #1,d0                   *
 58:            bgt     complp                  *
 59:
 60:            addq.l  #1,a2
 61: revlp:     move.b  -(a2),d1                *あとの処理に備えて
 62:            move.b  (a1),(a2)               *  パターンを反転しておく
 63:            move.b  d1,(a1)+                *
 64:            cmpa.l  a2,a1                   *
 65:            bcs     revlp                   *
 66:
 67:            move.w  -(a0),d0                *
 68:            addq.w  #1,d0                   *d0.l = パターン長
 69:
 70: cretn:     movem.l (sp)+,SAVREGS
 71:            rts
 72: *
 73: toolng:    moveq.l #-1,d0                  *エラーを返す
 74:            bra     cretn                   *
 75:
 76: *
 77: *          照合処理本体
 78: *
 79:            .offset 4
 80: txt:       .ds.l   1                       *テキスト
 81: txtend:    .ds.l   1                       *テキスト末尾
 82: *
 83:            .text
 84:            .even
 85: *
 86: bm_exec:
 87: SAVREGS =          d0-d5/a1-a5
 88: SAVSIZ  =          (6+5)*4
 89:            movem.l SAVREGS,-(sp)
 90:
 91:            movem.l txt+SAVSIZ(sp),a0/a5
 92:                                        *a0 = テキスト先頭
 93:                                        *a5 = テキスト末尾
 94:            lea.l   work,a1                 *a1 = ワーク先頭
 95:            lea.l   PAT(a1),a2              *a2 = パターン末尾
 96:            move.w  (a1)+,d1                *d1 = パターン長-1
 97:            bmi     qretn                   *パターン長が0だった
 98:
 99:            move.l  a0,d3                   *d3 = テキスト先頭
100:            adda.w  d1,a0                   *a0 = パターン末尾との重ね位置
101:            subq.w  #1,d1                   *d1 = パターン長-1のdbraカウンタ
102:            move.b  (a2)+,d4
103:
104:                               *パターン末尾と一致する文字を捜す
105:            moveq.l #0,d0                   *d0.wの上位バイトをクリア
106:            bra     next0                   *
107: loop0:     move.b  (a0),d0                 *d0 = 入力文字
108:            cmp.b   d4,d0
```

```
109:          beq     break0
110:          move.b  0(a1,d0),d0       *d0 = 対応する移動量
111:          adda.w  d0,a0             *ポインタを移動量の分進める
112: next0:   cmpa.l  a5,a0             *末尾を越えるまで繰り返す
113:          bcs     loop0             *
114:          bra     nomatch           *パターンは見つからなかった
115:
116: break0:  movea.l a0,a3             *a0 = パターン末尾との一致位置
117:          movea.l a2,a4             *a4 = パターン末尾の1文字手前
118:          move.w  d1,d2             *d2 = パターン長-1のdbraカウンタ
119:          bmi     zenchk            *パターン長が1だった
120:
121: loop1:   move.b  -(a3),d5          *パターン末尾側から順に比較する
122:          cmp.b   (a4)+,d5          *
123:          dbne    d2,loop1          *
124:          beq     zenchk            *全文字一致した
125:
126:          move.b  0(a1,d0),d0       *d0 = 不一致を起こした文字の移動量
127:          adda.w  d0,a0             *その分ポインタを進める
128:          cmpa.l  a5,a0             *末尾を越えたるまで繰り返す
129:          bcs     loop0             *
130:
131:                         *照合失敗
132: nomatch:moveq.l  #-1,d0            *N = 1
133:          bra     retn
134:
135:                         *照合成功
136: zenchk: ~
               :
               :
```

文字の出現頻度

本文でも少し触れたのだが，ある文字が文章中に現れる頻度は文字ごとにずいぶん違う。たとえば，ふつうの英文では“E”が最もよく使われ，“J”，“Q”なんかになるとあまり使われない。その使用頻度には何100倍とかいったレベルでの差がある。ここで，幅広くサンプルをとった統計を示せればよいのだが，あいにく引用できそうなものがすぐには出てこなかったので，代わりに手元にあった英文ファイルで文字の使用頻度を数えてみた結果を示しておこう(表A)。サンプルはMicroEMACSのドキュメントだ。大文字/小文字は同じものとして数えている。文中でMicroEMACSを連呼しているためか，少々“M”の使用頻度が高すぎるように思えたので，補助用に『GNU General Pablic License』での結果も並べておいた。こちらは短い文章なのであまり当てにはできないかとも思ったが，大筋では似たような結果になっている。

ついでに日本語でも同じようなことをしてみた(表B)。ひらがなだけを抽出し，濁点/半濁点の有無や，文字の大小(拗音や促音)は区別せずに数えた結果だ。適当なテキストファイルがなかったので，サンプルとしてはここ4回のこの連載の原稿を使っている。予想されるように，助詞や助動詞に使われる文字が上位を占めた。濁点を無視した関係で，濁音つきでも濁音なしでも助詞/助動詞によく使う“て”や“か”が最上位にきている。また，“の”は用途が広い助詞であり，指示代名詞にも含まれるから，この位置も頷ける。濁点の有無を区別するようにすれば，多分，トップになるだろう。もっとも，“の”が多用されているということは，それだけ文章に曖昧さや冗長さがあるということなのかもしれない。平均よりもずっと多かったりすると，ちょっと嫌だな。

ところで，この連載はこんな文体だ。「です•ます調」の文章で統計をとれば，“す”，“ま”あたりがもっと上位に食い込んでくるだろう。試してみようと思い，比較のためには自分の書いたものがよかろうと探したのだが，これまた適当なものがない（考えてみれば，ここ2，3年のあいだ，不機嫌さを隠す目的でしか「です・ます調」を使っていなかった)。ハードディスクを引っ掻き回し，アーカイブを覗きまくって，やっとのことで，2本ほど見つけることができた。結果は見てのとおり。期待どおりすぎてあまり面白くはない。

さて，どうでもいいようなこの文字の使用頻度というヤツは，意外にコンピュータとも関連が深い。たとえば，キーボードの文字配列(タイプライターが先だけど)。一般に使われているのはいわゆる「QWERTY配列」のものだが，よく使われる文字を打ちやすいキーに割り当てた「DOVRAK配列」は習熟が早く，また，長時間タイプしていても疲労が少ないという。QWERTY配列のほうは，そういったことがまったく考えられていない。というより，初期の機械式タイプライターでは，あまり速くキーを叩くとインクリボンを打つバーが絡まってしまうために，故意に速くは打てないように文字を配置したという話もまことしやかに伝わっている。

英文タイプに対して，JISのかなタイプの文字配列は，最初から文字の使用頻度がある程度考慮されているようだ。さっきの表Bと比べてみると，“ぬ”，“ふ”，“へ”，“む”なんかが端のほうに配置されているのも妙に納得できる。ホームポジション近くにはよく使われる文字が集まっているのもわかる。表Bと必ずしも一致しないのは，サンプルが悪いせいもあるだろうが，漢字を考慮しなかったのが効いているようだ。漢字を「読み」に展開して扱うと，また違った統計が得られるはずだ。そう考えてみると，JISのかなキー配列は“音読み”によく使われそうな文字がかなり優先的に配置されているのがわかる。

もっとプログラム寄りだと，暗号関係。弾道計算のつぎに現れたコンピュータの用途が，暗号の解読だったりしたわけだが，文字を別の文字に1対1で置き換える古典的な暗号では，文字の使用頻度を利用すると，比較的簡単に解読できるという。英文なら，最も多い文字が“E”だろうとあたりをつけるわけだ。ちなみに，英字の使用頻度のまともな統計結果が知りたければ，暗号関係の本を探すのが手っ取り早い。

あとはファイル圧縮だ。1文字単位で圧縮するアルゴリズムの場合，よく現れる文字は短いビット長で，あまり現れない文字は長いビット長で表すようなコード系に置き換えるということがよく行われる(そのようなコード系として有名なものにハフマンコードがある)。このようなコード系を得るために，圧縮ツールの中には文字の使用頻度の統計を内部に抱えたものもあると聞く。もっとも，通常の圧縮ツールでは圧縮しようとしているファイルについて統計をとって，ファイルごとに最適なコード系を決めるのがふつうなので，既存の統計を鵜のみにしても必ずしもよい効果は得られないかもしれない。が，ファイルを頭から読みながら，圧縮と並行してコード系をどんどん変化させていくようなプログラムの場合，その初期状態として既存の統計を利用するのも悪くない考え方だろう。

最後になったが，今回のテーマである文字列照合でも文字の使用頻度はある程度影響すると考えられる。特に，パターン先頭や末尾など，アルゴリズム上のポイントになる位置に置かれた文字は，より影響が強いに違いない。とすると，先頭1文字が非常に使用頻度が高く，2文字目が低いような場合，先頭ではなく2文字目をキーにして探すような変形も考えられる。もっとも，文字の種類が十分多い(つまり，ふつうの)状況では，その差はわずかでしかない。それでも，“0”と“1”だけからなる2進文字列を扱う場面で“1”のほうが圧倒的に多いことがわかっている，といった特殊な状況では，このような変形も意味があるだろう。

表A 英字の使用頻度(%)

1) MicroEMACS V3.9 のドキュメント

E	13.30	L	4.05	Y	1.59
T	8.49	H	3.72	B	1.54
A	7.31	D	3.58	V	0.95
N	7.15	M	3.38	X	0.82
O	7.06	U	3.14	K	0.69
R	6.97	F	2.93	Z	0.10
I	6.61	W	2.15	Q	0.09
S	5.95	P	2.13	J	0.06
C	4.55	G	1.68		

2) GNU General Pablic License (version 1)

E	11.68	C	3.72	W	1.67
O	9.22	D	3.31	B	1.45
T	9.02	U	3.05	V	0.96
R	8.10	L	3.05	K	0.35
I	7.55	P	2.96	X	0.21
A	7.38	F	2.78	J	0.04
N	6.03	M	2.65	Q	0.03
S	5.90	Y	2.63	Z	0.00
H	4.19	G	2.09		

表B ひらがなの使用頻度(%)

1) だ・である調

て	8.50	こ	2.20	お	0.55
の	7.52	り	1.98	ろ	0.48
か	5.97	も	1.86	や	0.39
る	5.92	ら	1.85	ほ	0.37
に	5.81	き	1.62	ん	0.35
と	4.87	よ	1.33	ひ	0.33
い	4.72	ま	1.29	へ	0.29
は	4.69	あ	1.27	ち	0.28
た	4.68	さ	1.15	せ	0.22
を	4.68	く	1.12	む	0.14
な	3.99	け	1.01	ふ	0.11
し	3.93	そ	0.84	ね	0.07
す	2.88	え	0.82	ゆ	0.04
つ	2.80	わ	0.72	ぬ	0.00
う	2.74	め	0.69		
れ	2.26	み	0.66		

2) です・ます調

て	8.72	つ	2.25	み	0.51
の	7.38	り	2.20	お	0.50
す	6.33	こ	2.02	や	0.48
か	5.36	も	1.90	め	0.44
に	5.20	ら	1.70	へ	0.36
と	4.99	さ	1.65	ひ	0.36
は	4.85	き	1.60	ち	0.36
ま	4.80	よ	1.46	ほ	0.35
い	4.17	あ	1.16	ろ	0.34
し	3.96	そ	1.04	ね	0.14
を	3.57	ん	1.00	ふ	0.12
な	3.43	く	0.95	む	0.09
た	3.13	け	0.94	ゆ	0.04
る	3.06	え	0.72	ぬ	0.01
れ	2.81	せ	0.69		
う	2.27	わ	0.60		

バックナンバー案内

ここには1992年3月号から1993年2月号までをご紹介しました。現在1991年1，5，9，11，12，1992年1，6～12，1993年1，2月号の在庫がございます。バックナンバーおよび定期購読の申し込み方法については166ページを参照してください。

1992

3月号（品切れ）
特集　SCSIの活用

連載　響子 in CGわ～るど/DōGA CGA/大人のためのX68000/Z80's Bar
ショートプロ/吾輩はX68000である/マシン語プログラミング
ハード工作/ANOTHER CG WORLD/Computer Music入門/カードゲーム

●Z-MUSIC支援ツール　ZPDCON.X
●Z's-EX用拡張コマンド　MASK_reverse.X
LIVE in '92　ギャラクシーフォース/君が代
THE SOFTOUCH　グラディウスII/レミングス/大戦略III'90/伊忍道
全機種共通システム　カードゲームKLONDIKE

4月号（品切れ）
特集　成熟するゲームと日本の文化

連載　よい子のSX-WINDOW/Z80's Bar
響子 in CGわ～るど/ショートプロ/吾輩はX68000である
ハード工作/ANOTHER CG WORLD/Computer Music入門

●発表　1991年度GAME OF THE YEAR
●バーコードバトラー
LIVE in '92　あじさいのうた/ショパン練習曲作品25-2へ短調/IT'S MAGIC
THE SOFTOUCH　ファーストクィーンII/マスターオブモンスターズII 他
全機種共通システム　実践Small-C講座(1)オプティマイザ080

5月号（品切れ）
特集　明日のための環境づくり
第7回　言わせてくれなくちゃだワ

連載　響子 in CGわ～るど/大人のためのX68000/Z80's Bar
ハード工作/ショートプロ/マシン語プログラミング
Computer Music入門/吾輩はX68000である

●製品紹介　MIDI音源 03R/W/MIC68K
LIVE in '92　フレンズ/Danger Line
THE SOFTOUCH　エイリアンシンドローム/苦胃頭捕物帳 他
全機種共通システム　実践Small-C講座(2)COMMAND.OBJ

6月号
特別企画　Oh!MZ，Oh!X10年間の歩み
特別付録　創刊10周年記念PRO-68K(5"2HD)

連載　響子 in CGわ～るど/大人のためのX68000/マシン語プログラミング
ハード工作/ショートプロ/ANOTHER CG WORLD/Z80's Bar
吾輩はX68000である/Computer Music入門

●新製品紹介　Z'sSTAFF PRO-68K ver.3.0
LIVE in '92　Shake the Street/Ancient relics
THE SOFTOUCH　スピンディジーII/ロイヤルブラッド/ライフ&デス 他
全機種共通システム　実践Small-C講座(3)COMMAND.OBJ2

7月号
特集　超空間美術論
特別付録　DōGA CGAシステム&お試しディスク(5"2HD)

連載　よいこのSX-WINDOW/響子 in CGわ～るど/Z80's Bar
ANOTHER CG WORLD/大人のためのX68000
Computer Music入門/ハード工作/ショートプロ

●試用レポート　V70アクセラレータボード
LIVE in '92　Bye Bye My Love/MATERIAL GIRL/ヴェクザシオン
THE SOFTOUCH　将棋聖天&棋太平68K/シムアース/太閤立志伝
全機種共通システム　実践Small-C講座(4)関数リファレンス

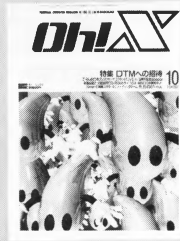
8月号
特集　プログラミング再入門

連載　響子 in CGわ～るど/吾輩はX68000である/よいこのSX-WINDOW
マシン語プログラミング/ハード工作/ANOTHER CG WORLD
大人のためのX68000/Computer Music入門/ショートプロ

●新製品紹介　MATIER/TG100/SOUND SX-68K
LIVE in '92　氷穴/ガラガラヘビがやってくる/風の贈り物
THE SOFTOUCH　三國志III/シムアース/ウルティマVI/バトルテック
全機種共通システム　実践Small-C講座(5)ワイルドカード
グラフィックライブラリGRAPH.LIB

9月号
特集　数値演算の熱い逆襲

連載　DōGA CGアニメーション講座/大人のためのX68000
響子 in CGわ～るど/吾輩はX68000である/ショートプロ
マシン語プログラミング/ハード工作/ANOTHER CG WORLD

●新製品紹介　MATIER/MIREGE Model Stuff
LIVE in '92　恋をしようよ Yeah! Yeah!/ゆめいっぱい
THE SOFTOUCH　ファイナルファイト/ライジングサン/ヨーロッパ戦線/シューティング68K GAMES
全機種共通システム　O-EDIT & MODCNV

10月号
特集　DTMへの招待

連載　DōGA CGアニメーション講座/大人のためのX68000
響子 in CGわ～るど/吾輩はX68000である/ショートプロ
マシン語プログラミング/ハード工作/ANOTHER CG WORLD

●試用レポート　X68000用CD-ROMドライブ
LIVE in '92　美少女戦士セーラームーン/笑顔を探して 他
THE SOFTOUCH　ポピュラスII/リーディングカンパニー/ネクタリス/サークII
全機種共通システム　実践Small-C講座(6)SLENDER HUL

11月号
特集　ゲームマネージメント

連載　DōGA CGアニメーション講座/大人のためのX68000
響子 in CGわ～るど/ショートプロ/よいこのSX-WINDOW
ハード工作/ANOTHER CG WORLD/Computer Music入門

●新製品紹介　CHART PRO-68K
LIVE in '92　ストリートファイターII/スーパーマリオ 他
THE SOFTOUCH　キャッスルズ/シュートレンジ/ポピュラスII/サンダーレスキュー
全機種共通システム　実践Small-C講座(7)EDIT

12月号
Oh!X5周年特別企画　ショートプロ大集合

連載　DōGA CGアニメーション講座/マシン語プログラミング
響子 in CGわ～るど/ショートプロ/よいこのSX-WINDOW
大人のためのX68000/ハード工作/Computer Music入門

●エレクトロニクスショウ'92
LIVE in '92　LAST CHRISTMAS/闇の血族/ユーフォリー
THE SOFTOUCH　デスブレイド/ムーンクレスタ&テラクレスタ/ふしぎの海のナディア/ロードス島戦記II 他
全機種共通システム　実践Small-C講座(8)MAKE

1993

1月号
特集　D.I.Y.ハードウェア

連載　DōGA CGアニメーション講座/マシン語プログラミング
響子 in CGわ～るど/ショートプロ/よいこのSX-WINDOW
大人のためのX68000/ハード工作/Computer Music入門

●新製品紹介　サンダーワード/SX広辞苑
LIVE in '93　ムーンライト伝説/チャコの海岸物語
THE SOFTOUCH　オーバーテイク/ストライダー飛竜/エアーマネジメント/パイプドリーム 他
全機種共通システム　実践Small-C講座(9)EDC-Tの拡張

2月号
特集　画像創造のために

連載　DōGA CGアニメーション講座/マシン語プログラミング
響子 in CGわ～るど/ショートプロ/よいこのSX-WINDOW
ハード工作/吾輩はX68000である/Computer Music入門

●新製品紹介　Communication SX-68K
LIVE in '93　FIRE CRACKER/サンバDEグワッシャ!
THE SOFTOUCH　極/ドラゴンスレイヤー英雄伝説/機甲装神ヴァルカイザー/キングス・ダンジョン
全機種共通システム　BLACK JACK

THE USER'S WORKS

●ふぁ〜すとくらいしす/ProstituteMaker

今回紹介するのはお馴染みT&H PROJECTSの新作だ。「クイズ」と「子育て」，２本ともかなり完成度の高い作品となっている。

●ふぁ〜すとくらいしす

このゲームは戦闘の代わりにクイズを使ったRPGだ。フィールド画面では2D，ダンジョン内では3D表示でゲームが進行する。このあたりは前作(？)デースレ３の雰囲気も備えている。ついでにデースレ３に出てきたジーナがお助けキャラクターとして登場する。機嫌のいいときには問題の答えやヒントを教えてくれるという設定だ。

クイズは４択形式で行われる。全体的なクイズの難易度はなんともいいがたいところがある。問題はゲーム，アニメ，特撮，ミリタリー関係の問題が多いようだ。制限時間がないのも特徴といえるだろう。

こういった類のゲームはクイズ自体の難度よりも問題数のほうが大きく難易度に影響する。ディスクを見ると175Kバイトのクイズデータらしきファイルがある。全角文字に換算すると，Oh!Xを文字だけで埋めて30ページ分のデータ量に相当する。しかし，その割には何度も同じ問題を見ている気がするが……。出題の際に前に出た問題をチェックしていないのでこのようなことになるのだろう。それを利用した引っ掛け問題もあるので，これはこれでいいのかもしれない。

RPG風ゆえ途中でアイテムも使用できる。アイテムには体力を回復するもの，わからない問題をスキップするもの，問題のヒントや答えを教えてくれるもの，一時的に攻撃力や防御力を上げるものなどがある。それぞれ有効回数が決まっており，携帯できるアイテムの最大個数も決まっている。主装備は自動的に装備されるので後半になるとかえって難易度は下がってしまう。前半に「敵のHPが60でこちらの与えられるダメージは１」という展開になったときは非常に燃えたのだが……。

一部のクイズに怪しい点もあるが，全体的にかなりよくできているといっていいだろう。マップが小さいことが少々残念である。クイズの数ももう少しほしかったところ。続編も予定されているようなので，期待しよう。

かるーいノリの「ふぁ〜すとくらいしす」。ちょっと見ると普通のPRGなのだが，戦闘はクイズで行う

●Pメーカー

写真を見ればわかるとおり，これは某有名作品をややH版にしたものだが，こういったエセソフトに疑問に持つ方もいるだろう。焦点は「どこまで許されるか」だ。システムまわりは著作権の範囲外(特許関係かな)，グラフィックと音楽は書き起こし，基本アイデアはどちらもプロダクションマネージャの簡易版といった感じ。が，安易な真似はほめられたものではない。T&Hの場合ちゃんと許可を得て販売しているというのは立派なものだ。

内容を解説しよう。巨大歓楽街を抗争から救った男に街のドンから褒美が取らされることになった。男は憧れのヒモ生活を実現するため女の子を……という設定で物語は始まる。

５年間かけて娘を育てるわけだが，娘が更生してしまうと風俗関係のバイト（実入りがよい）をしなくなるので適度にグレた娘に育てるのがよい。こういうと倫理的に抵抗を感じる人もいるかもしれないが，職業に貴賤なし，それもまた人生と割り切るのが正しい。

元になる作品があるということで，よいところも悪いところも受け継いだという印象を受ける。ゲームバランスは改善されているとも取れるが，展開が単調になりがちだという点は否めない。多彩なエンディングを生かすにはイベントが足りないせいだろうか。

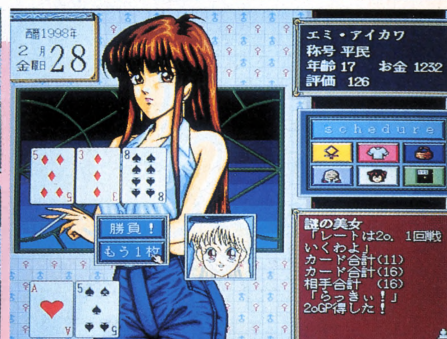
技術的にはほとんど問題はなく造りも丁寧。当然プログラムはX68000用に作られている。絵は256色で音楽は８声，女の子はしゃべるという仕様だ。ディスクは２枚しかないがグラフィックが極端に少ないということはない。

●入手方法

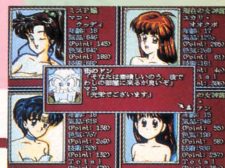
郵便局で代金分の定額小為替を買って，無記名のまま，希望するソフト名を書いた紙，返送用の宛名シールを同封して，

〒560　大阪府豊中市本町8-6-28
T&H PROJECTS豊中支部

まで連絡を。価格は以下のとおり（送料込み）。

ふぁ〜すとくらいしす　1200円
ProstituteMaker（要2Mバイト）1800円

「Pメーカー」による13歳から17歳までのグラフィック例。やはりあれを押すとあんな絵も出るものの，それほど「Hゲーム」なわけではない。しかし，この手のゲームを略すとみんなP.M.になるのはなぜだろう

ミスP嬢コンテスト。最大のイベントである

かくしてこの少女の受難の日々が始まる……

感動のエンディング。５年の月日は運命をこうも変える

［第22回］
狐の顔

寺尾　響子

響子inCGわ〜るど

改札口を出てみると，最終バスはちょうど出たあとでした。その日にかぎって駅前にタクシーもなく，歩いて帰ることにしました。会社で終わらなかった仕事を詰めたカバンが，いよいよ重く感じられます。我が家まで歩いて30分。でも，稲荷神社のあった空き地を抜ければ5分は稼げるに違いない。よし。

暗い空き地に差しかかりました。小さい頃，よく遊んだ場所です。

あんまり遅くまで遊んでいると神社の狐がどこか遠くへ連れていってしまうよ……
なにをいっているんだばあちゃん
狐なんかこんな町中に出るわけねえじゃんかぼけるには早いぞ……

祖母の親身な心配も素直には受け取れません。塾をさぼって日が暮れるまで遊び，暗くなればやはり祖母の狐の話を思い出し，怖くなって境内の石段を大いそぎで駆け降りてゆくのでした。あたたかい晩ごはんを目指して。

そのうち，地域振興のための再開発とやらで稲荷神社はどこかへ移され，大型のゲームセンターができました。私たちの遊びは境内でのプロレスごっこやサッカーから，室内での格闘技ゲームやシューティングゲームに。

そのゲームセンターもすでに取り壊され，あとには空き地と石段だけが残りました。いまでは遊ぶ子供の姿もありません。

うらうらと思い出に浸っているそのとき……。

こん！

狐が目の前に現れたのです。いや，正確には大きな狐の顔でした。立体映像かホログラムのような。ときどき透きとおってゆらゆらと揺れるようすは，かげろうのようでもありました。それが，こんな歌を口ずさむのです。

帰ろよ帰ろ　いっしょに帰ろ

あの頃へ　黄金のコンピュータ時代へ
ゲームセンターへ　アキハバラへ
帰ろよ帰ろ　子供に帰ろ

やれやれ，疲れたせいで見る幻かな残業が多いからなこのごろ景気が悪いのにやたら雑用はあってしかも残業代は出ないんだもんな……

まばたきをし，じっと真正面を見据えました。しかし，狐の顔はあいかわらずそこにあり，右に左に大きく揺れてとおせんぼをするのでした。

帰ろよ帰ろ　　いっしょに帰ろ
子供に帰って　いっしょに遊ぼ　こん！

ぞっとしました。このまま連れ去られてしまうのではないかと思いました。年をとった私を連れていってどうするのでしょう。

遠くに灯りがぼうっと浮かびました。我が家の確かな光。ふんわりと柔らかな香りが漂ってくるような気がしました。晩ごはん。妻の手料理。胸いっぱいにその香りを吸い込んだつもりになって，息をこらえました。

ごめんよごめん君とは遊べないんだよ遊ぶにはもう遅すぎるんだよ……

狐の顔が崩れました。悲しい目をして透明になってゆきました。

ほんとにごめんまたいつかきっと……

それからカバンを抱きかかえ，狐の顔を突っきって石段を駆け降りました。

製品紹介

MIRAGE System Model Stuff

Tan Akihiko
丹　明彦

「MIRAGE System Model Stuff」がマイナーバージョンアップした。大きな変更点，機能強化はないが，バグはほとんど解消。使いやすい環境を手軽な価格で実現できるソフトだけに，メーカー側の配慮は心強い。

「MIRAGE System」はモジュール拡張型3DCGシステムである。「Model Stuff」はそのシリーズ第1作で，主にモデラの機能を提供する。

初期バージョンの発売から数カ月が経過した昨年の暮れに，細かいバグ等を修正したバージョンが供給されたので，ここに追加レビューを行う。

なお，前回のレビュー（1992年9月号）で紹介したバージョンと基本的には同じバージョンなので，細かいスペックに関してはそちらもご参照いただきたい。

Model Stuffの概要

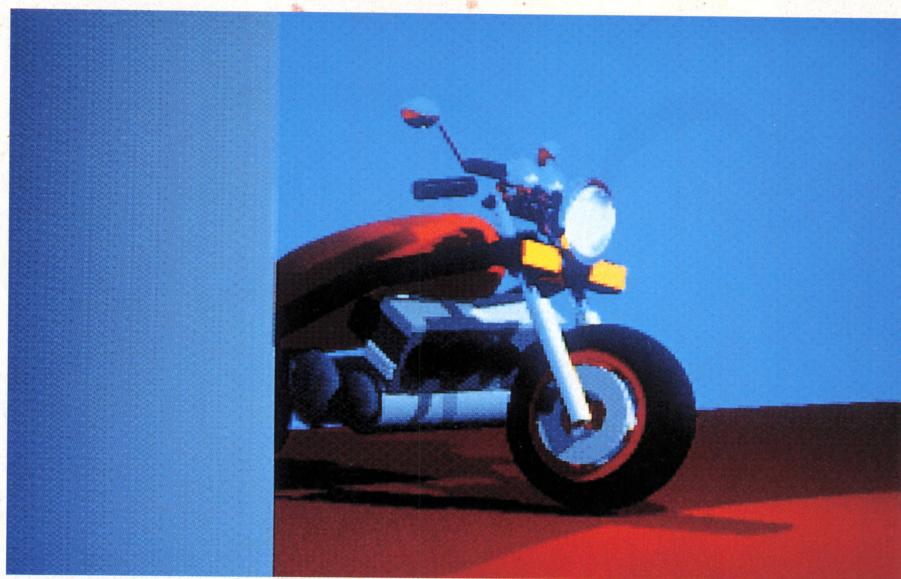
製品の内容は，

○モデラ
○（簡易）アトリビュータ
○レンダラ
○MIRAGEシェル

からなる。

モデラは，プリミティブ（基本立体）やその組み合わせであるマクロによって，シーンを対話的に作成する。

アトリビュータは，作成したシーンに色や反射率，屈折率などの属性（アトリビュート）を付加する。

X68000用　5″2HD版2枚組　29,800円（税別）
メディックス　☎03（3950）2222

前回使ったオートバイ。レンダラのバグが取れた　レンダリング時間　15時間46分48秒（XVI，コプロなし）

レンダラは，作成したシーンのレンダリング（描画）を行う。レンダリングのアルゴリズムはレイトレーシングである。レンダラは，数値演算プロセッサがあるマシンとそうでないマシンのそれぞれに対して，より高速なバージョンが用意されている。

MIRAGEシェルは，メニュー形式でモデラやアトリビュータ，レンダラ，それに画像ビュアやペインティングツール（「Z'sSTAFF」と，今回のバージョンからは「MATIER」も）を呼び出せる。CGを作成するための統合的な環境を提供しているわけだ。

マクロを使う

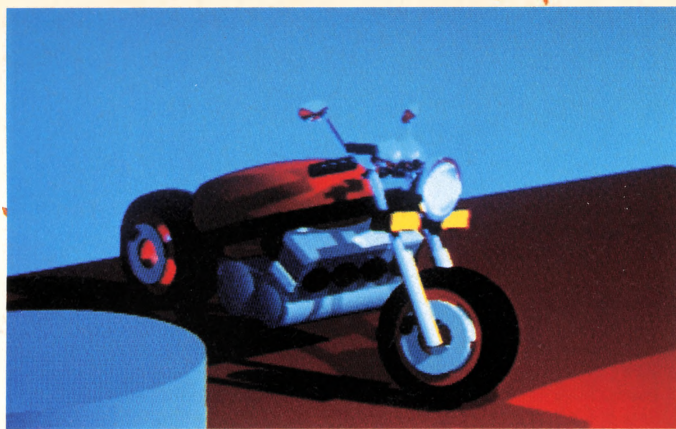
前回あまり詳しく触れなかったマクロである。マクロの機能はプリミティブを寄せ集めて，ひとかたまりの物体として扱うことであるが，マクロはまた論理演算をするための単位ともなる。

作例のサングラスは，論理演算を行ってみたものである。ポリゴンは使っていない。楕円柱5個と直方体1個で論理演算している。ちょっと慣れればこのくらいは数分でモデリングできる。

また，「Model Stuff」の特徴的な機能として，プリミティブおよびマクロの回転中心を指定することが挙げられる。

サングラスはモデリングしたままの直立した状態では変なので，レンズがテーブルと接触している点を回転中心に指定し，その周りをサングラスのつるの先がテーブルにつくまで回転させる。このとき，マクロごと回転していることにも注目しておこう。この参照回転機能はおいしいぞ。操作性もいいし。

さらに応用として，マクロと参照回転機能を酷使する「多関節キャラクタ」を作ってみた。

まず手である。手は掌と5本の指とからなる。人差し指，中指，薬指，小指については，掌に接続する部分も合わせて関節が3つある。第1関節から先をひとつのマクロ，第2関節から先もマクロ，第3関節も同様というぐあいに，多重階層構造をもたせたマクロを構成した。親亀の上に子亀を，

上から見るとモデリングの手抜きがばれる

反射率マッピングで金属の質感を表現

のノリである。

指は1本作ってしまえば，マクロのコピー＋拡大縮小でけりがつく。これで4本の指は完成し，これに掌と，同様の多重構造で作った親指をくっつければ手は完成だ。

あとは同様で，手首，前腕，肘，上腕，肩と作って，ここまでを腕とする。マクロの階層も深くなったような気がするが，これでもまだ7～8階層程度。システムは32階層までサポートしている。

腕をコピーして右腕と左腕にし，胴体と頭を作る。これで上半身が完成。下半身は同様ということで作らなかった。

次はこの多関節キャラクタを動かす。ここで参照回転機能の真価が発揮される。

マクロごとに回転中心を設定していく。そのマクロが表現する部位のすぐ上の関節(たとえば前腕部なら肘，上腕部なら肩)の中心を指定する。マウスでクリックするだけなので簡単だし，なにより目で見て指定できるのがうれしい。

そしてマクロを回転させる。肩から先，肘から先がいっせいに動く。

いったんマクロに指定した回転中心はずっと記憶されているので，あとからポーズを変えたいと思ったときでも大丈夫。

質感表現

前回のレビューで完全に失念していたのが，金属の質感である。「MIRAGE System」はレイトレーシングそのものの機能はごくオーソドックスなものであるが，特筆すべきものに「反射率マッピング」というものがある。

金属に顕著にみられる性質として，反射率が角度によって複雑な分布をするというものがある。「MIRAGE System」では，これを導入することによって，ごく簡単な操作で金属の質感を出すことに成功した。

写真で反射率マッピングを施したものとそうでないものとを比較すれば，その効果はあきらかであろう。

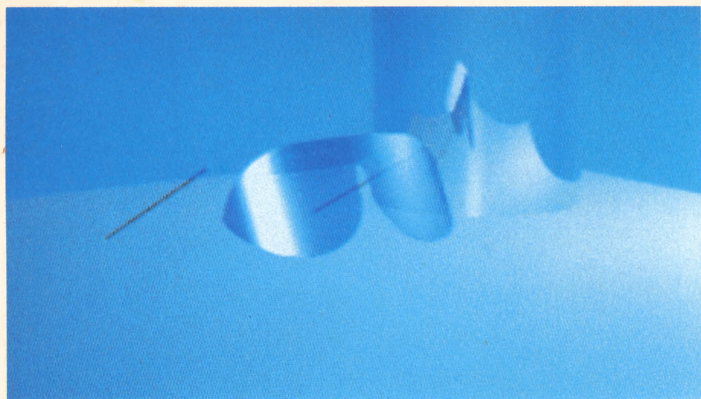
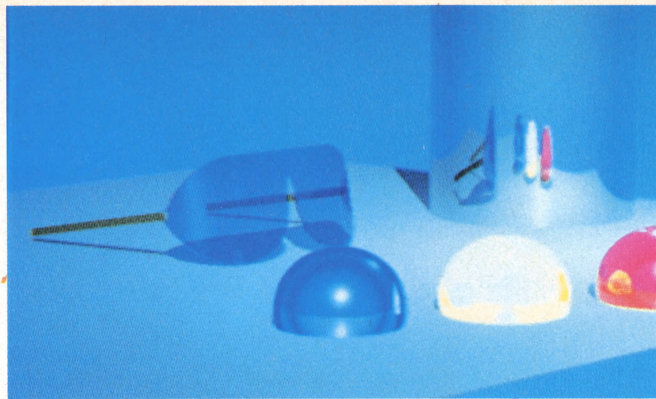
この反射率マッピングは，現バージョンでは金，銀，アルミニウム，ステンレスのデータが用意されている。

ポリゴンを使う

前回できなかったなどとグチっていたところである。ポリゴンモデリング/レンダリングツールである「Z'sTRIPHONY」でデザインしたポリゴンデータを，付属のコン

Z'sTRIPHONYからコンバートしたポリゴン

同じくポリゴン。金属製の折り鶴

サンプルでついてくる画像。水飲み鳥

サングラスの制作途中。後ろは金属柱

サングラス。球には反射率マッピング　4時間47分21秒

MIRAGEシェル。いろいろなツールを呼び出す

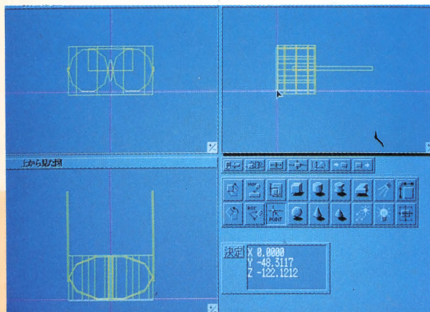
バータで「MIRAGE System」上に持ち込むことができる。現バージョンでは形状そのものの操作はできない。アトリビュータで色の設定を行い，レンダリングする。もちろん，通常のプリミティブやマクロとの混在も可能である。

コンバート作業に手間がかかることを除けば，実に有用な機能といえる。

安心して使えます

今回のバージョンを使ってみた印象を簡単にまとめてみる。

・モデラの信頼性が格段に向上している。たとえばプリミティブの削除も安心して行える

・初期バージョンに見られたレンダラのバグも取れ，ポリゴンなどのレンダリング結果が予期したものと違うということがなくなった

・そのほか，細かいバグがほとんどなくなっている

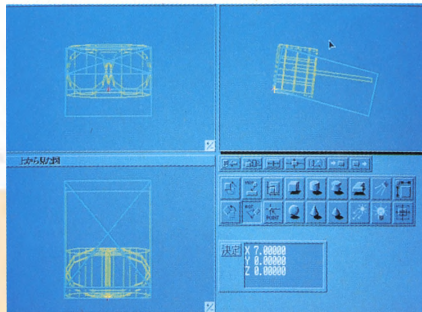
というわけで，特に新しい機能が追加されたわけではない（MIRAGEシェルから「MATIER」を呼び出せるようになったくらい）が，安心して使えるようになっている。これは大きい。同時に，きちんとサポートしていくという姿勢も示したことになる。これまた頼もしい話だ。

半面，初期バージョンが抱えていると思う大きな問題（速度や操作性）についてはそのままである。速度については，不満なら速いマシンを待つか，PC-9801を買えば解決するので問題ない（後向きの方法ではあるが)。操作性は，エンドユーザーに徹しようとすると細かいところで気になる点が多いというだけで，3DCGがわかっていればあまり問題ないだろう。

競合ソフトと比較して

X68000向けに市販されているレイトレーシングソフトとしては，

○C-TRACE（C-TRACE＋）　キャスト
○サイクロン（サイクロンExpressα）　アンス・コンサルタンツ
○MIRAGE（MIRAGE System Model Stuff）　メディックス

を挙げることができる。いずれもシリーズ化されているわけだが，最上位バージョンで代表して，これらを簡単に比較する。

1）　機能

まず扱える形状だが，三者ともにプリミティブとして2次曲面と直方体を扱える。これらはレイトレではお約束といえる。

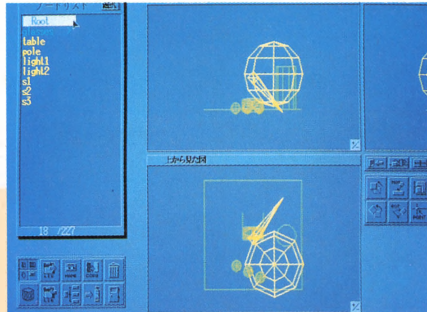
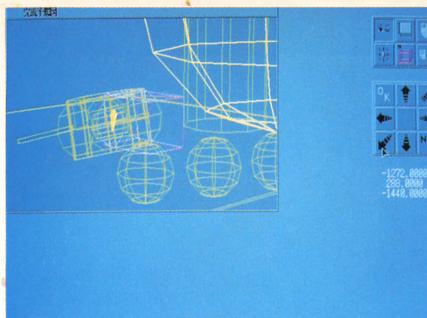
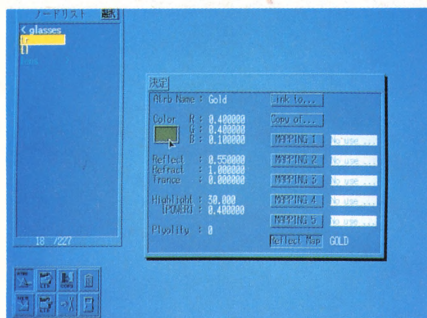
このほかに，「C-TRACE」ではメタボールを扱える。「サイクロン」と「MIRAGE」では，「Z'sTRIPHONY」からコンバートしたポリゴンデータを扱うことができる。使い勝手はどっこいどっこいで，コンバートの手間は同じくらいかかる。ただし「MIRAGE」は，新システム（Poly Stuff）の追加により状況が改善される見込み。

プリミティブの間の論理演算，グループ化（「サイクロン」や「MIRAGE」ではマクロ，「C-TRACE」ではクラスタと呼ぶ）によるツリー構造のサポート，それにプリミティブ単体またはグループ単位の移動および回転ができることも共通している。

光源は平行光源，点光源，スポット光源。これも共通。ただし「MIRAGE」は光源をプリミティブと同等に「物体」として扱っており，マクロに組み込んで自在に動かすことができるなどの優秀さを見せている。

アトリビュートの表現力は三者ともほぼ同等。テクスチャマップ，バンプマップ，アトリビュートマップといったところは押さえている。さらに「MIRAGE」では反射率マップをサポートしており，金属の表現

レンズが机に接触する点を中心にして……

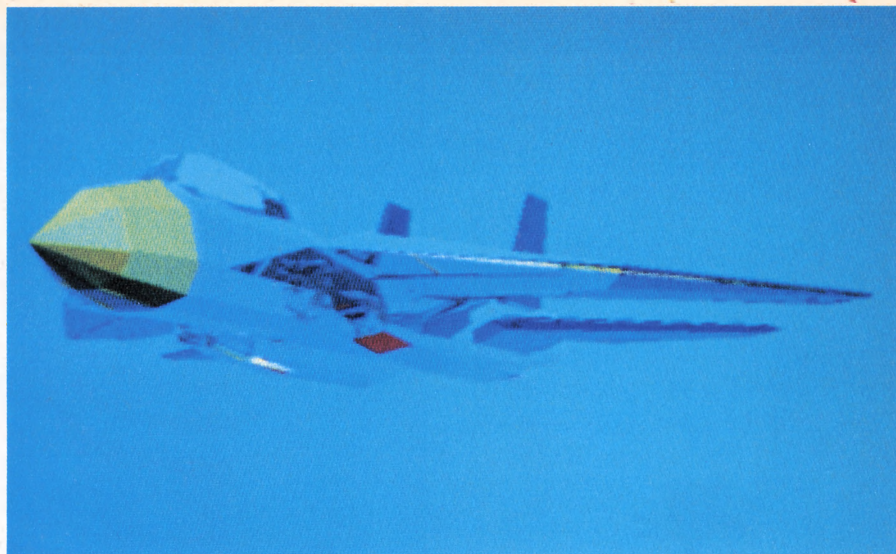
つるの先が机につくように回転させる

三面図モードでは編集を行う

透視図モードでは構図を決める

アトリビュータで反射率マッピングを設定

においてポイントを稼いでいる。

2) ユーザーインタフェイス

「C-TRACE」にはモデリングツールがない。シーン定義ファイルのフォーマットを公開していて，ユーザーがテキストエディタを使って記述することになる。ただし，ワイヤビューおよびメタビューにより，レンダリングする前に確認する程度のことは可能。が，論理演算はしてくれない。

「サイクロン」と「MIRAGE」には専用のモデラが付属している。サイクロンがキーボードオペレーションだったのに対し，MIRAGEはマウス指向。画面を見ながらインタラクティブにシーンを定義するのにはマウスのほうが適しているというのは，改めて指摘するまでもないだろう。

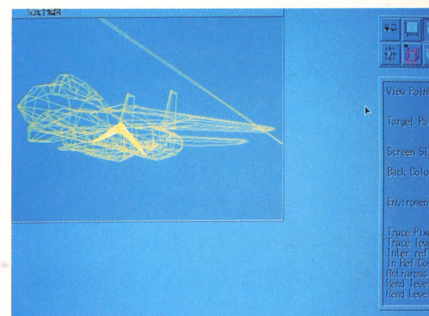
いずれも，ワイヤフレーム表示では論理演算はしてくれない。

Z'sTRIPHONYからコンバートした戦闘機

3) 速度

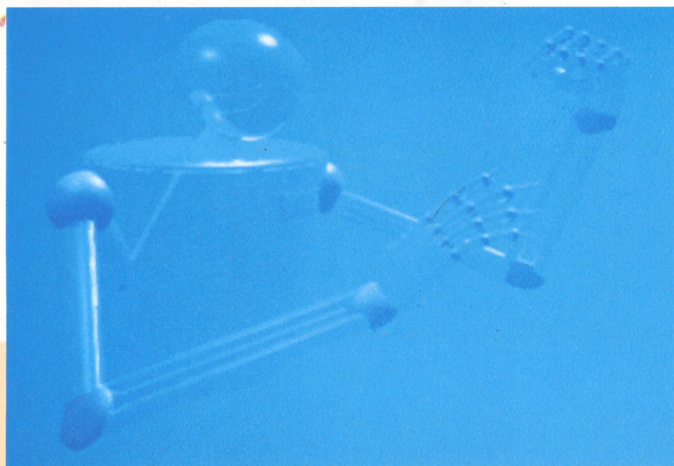
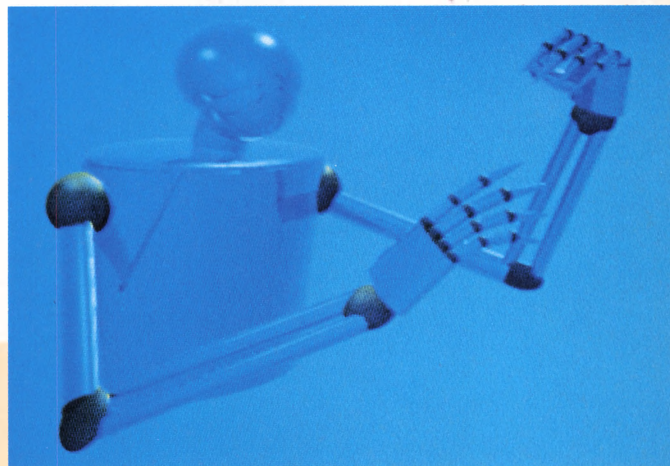
レンダリング速度は単純比較できないので，ある程度主観的になってしまうことをお許しいただきたい。

まず，「C-TRACE」にはトランスピュータバージョンが存在する。「サイクロン」にも存在する。このトランスピュータバージョンは，そもそもCPUが違うものを使っているのだから文句なく速い。最近安くなったとはいえ結構な値段なので，プロの使うものという気がしなくもない。

では，トランスピュータのない場合はどうかというと，三者ともボクセル分割による高速化アルゴリズムを採用していて，本質的な速度差はない。X68000においては，数値演算プロセッサは劇的というほどには速度向上をもたらさない。

「C-TRACE」は部分レンダリングにより，全体を粗くレンダリングして詳細な情報がほしい部分だけを細かくレンダリングすることができる。いうまでもなく，正直に全

透視図モードで描画パラメータも設定する

ピクセルを計算するのに比べて時間を節約できる。「MIRAGE」の場合は，最初は粗く，しだいに細かくというようなレンダリングの進め方をするので，あきらかなモデリングの間違いやアトリビュート指定の間違いを早期に発見することができ，また放っておけば自動的に細かいレベルのレンダリングまでやってくれる。個人的には「MIRAGE」の作法が使っていて気持ちいい。

レンダリング中断/再開の機能は三者に共通である。

反射率マッピングなしでレンダリング　2時間46分41秒

反射率マッピングつき。質感の違いに注目　3時間33分16秒

4) **価格**（すべて税別）

C-TRACE+	198,000円
C-TRACE TP+ （トランスピュータボード込)	398,000円
サイクロンExpressα	98,000円
MIRAGE System Model Stuff	29,800円

というわけで,「MIRAGE」が驚異的に安い。とはいえ「MIRAGE」にも多少高価でもいいから，トランスピュータ版のサポートはほしいところ（当然，発売予定には挙がっているだろうけど)。これがないと,「C-TRACE」や「サイクロン」のユーザーをひきつけることは難しいかもしれない。

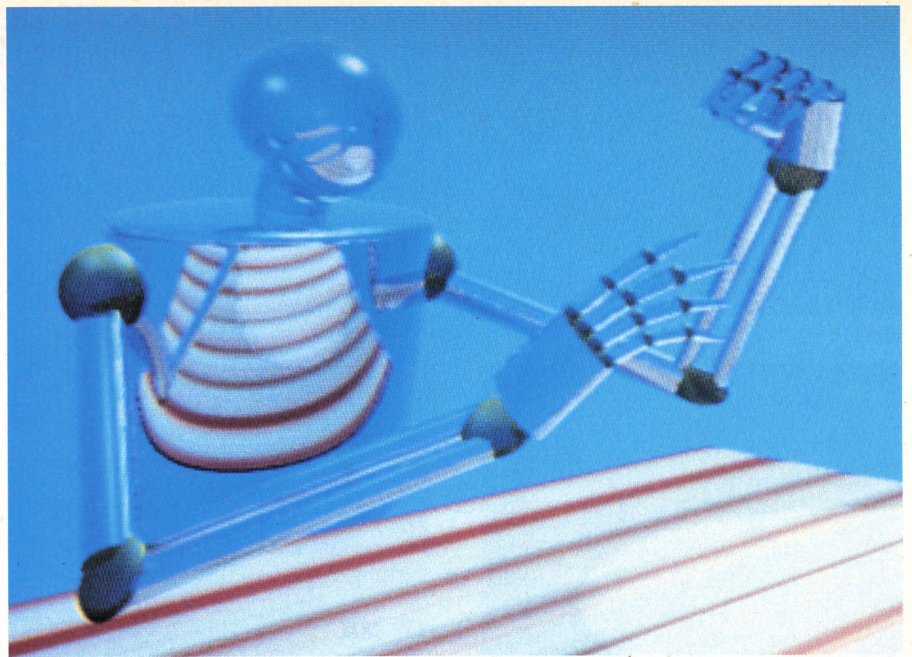
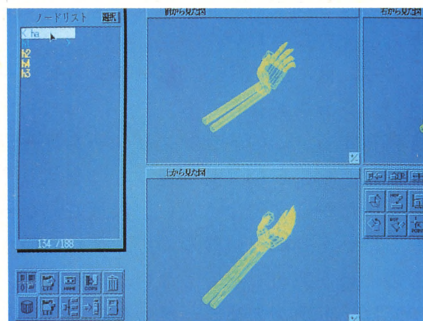
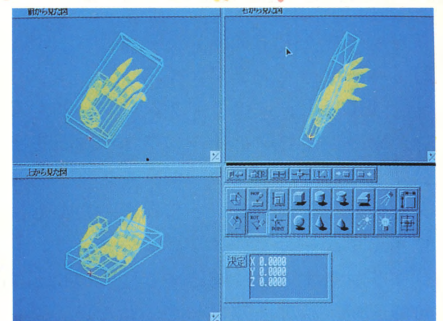
終わりに

なんだか上の比較を見ると,「MIRAGE」に有利になってしまっているが，標準的な機能で比べると後発の強みが出るということにすぎない。最後には厳しいこともいわせていただく。

いまのところMIRAGEに対して切実にほしいと思っているのが,

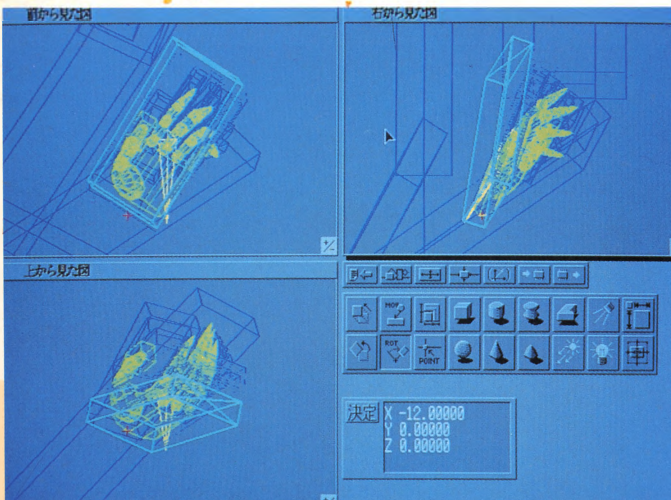
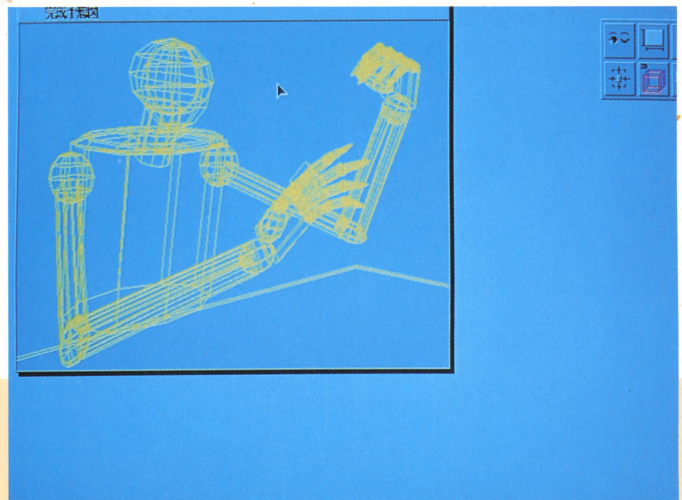
・スムーズシェーディングのポリゴン
・スプライン曲面でのオブジェクト作成
・メタボール

のいずれか，すなわちいわゆる自由曲面である。

テーブル（市松模様が望ましい）の上に球（材質は赤いプラスチックか金属，もしくはガラスが望ましい）を置く。不自然なまでに透明な空間の中でキラキラと輝く画像。こういう画像は典型的な「レイトレくさい」画像であり，一部からレイトレが不評を買う原因にもなっているのであるが，これを克服するのがレイトレの大きな課題だと思うのだ。

映り込みを強調するために机を置いてみた　5時間7分57秒

肘から先がひとつのマクロになっている

手首を中心に手のマクロを回転させる

曲面の映り込みを緻密に表現できるのがレイトレの強みである。球などの2次曲面でそれはよくわかった。次は表現力だ。フラットシェーディングのポリゴンでは役不足だろう。

＊　＊　＊

前回のレビューの，安定してきたら云々というくだりにあらぬ不安を抱かれた方も，今回のバグフィクスにより問題点は解消された。「MIRAGE System」は，現時点でのX68000におけるレイトレの環境としてはいちばんまともであるといっていいと思う。また，これからシステムの追加などでどんどん機能的にも充実していくはずである。

ほかのマクロを表示させて目安にする

ポーズをつけるのも簡単だ

[特集]

X-BASICを学ぶ

BASICとX-BASIC，そしてC言語。これらのあいだに特に明確な絆というものは見あたらない。X-BASICはむしろ独自の言語仕様を持っていると考えたほうが理解しやすいときもある。それでもたまにC言語の知識を要求されることがある。X-BASICは新しいタイプの高級言語である。
X-BASICがもっとも使いやすい言語だなどというつもりはない。不自由なところもあれば，不備もある。言語仕様を少し拡張するだけでもっと使える言語になるのにと思うことも少なくない。しかし，我々の前にはとりあえずX-BASICがあった。そして，それは，そう，悪いものでもない。高性能なコンパイラと組み合わせることによって，思わぬ高性能言語に早変わりすることすらある。
基本的にプログラミングという作業の本質はプログラミング言語に依存するようなものではない。基本的にアルゴリズムからコーディングに至る過程はどんな言語でもさほど変わらないものだ。
頭の中のイメージをより直接的にプログラム化する，そんな，「簡単さ」がBASICの美点である。とりあえず触ってみる。そこからしかX-BASICへの道は始まらないのだから。

[CONTENTS]

BASIC

プログラミングスタイルから見た

X-BASICと関数

Nakano Shuichi 中野 修一

X-BASICでのプログラムはほかのBASICに慣れた人にはとっつきにくい点もあるようです。ここではX-BASICの特徴的な関数の使い方とそれによるプログラミングスタイルについて探ってみましょう。

とりあえずX-BASIC

できあいのアルゴリズムを引っ張ってきて使うだけならプログラミングはコーディング主体の作業ですが，私も文系人間なので便利なアルゴリズムには暗いほうなのです。で，しかたがないのでたいてい自己流で片をつけてしまいます。

こういってはなんですが，なにかちゃんとした完成品を作る場合にはBASICは不向きな言語です。もっと実行時の効率を追求したほうがよいでしょう。こういうときにはバキバキのアセンブラが美しいですね。

しかし，試作品を作るときにはBASICはなによりも手軽にできます。プログラミングでは，やはりアルゴリズムを作ること，「どうやったらできるだろうか」と，いろいろ考えている時間が楽しいのでしょう。動くかどうかわからないアルゴリズムを試すときには最低限暴走の危険のない環境が必要です。68000ではある程度，暴走は起こりにくいとはいえ，飛ぶときはあっさり飛びます。これはアセンブラだろうがC言語だろうが似たようなものです。

X-BASICは難しい?

いつの世にも初心者というものはいるもので，いまだに「X-BASICは難しい」という声を聞くことがあります。

X-BASICというものを見た場合，簡単にいくつかの特徴を挙げることができます。

C言語にコンバートできる
関数を定義できる
ローカル変数が使える
行番号が使いにくい

ざっとこんなもんでしょうか? いにしえのBASIC自体がFORTRANの流れから派生したものであるにもかかわらず，X-BASICはありがちなALGOL系の言語に仕上がっています。きょう日のまともなプログラミング言語の9割はALGOL系といっても過言ではありませんから，これは特に「難しい」ということにはならないでしょう。

FORTRANに比べALGOLやPASCALはより「高級」であると表現されます。これは，より「人間にわかりやすい」ということを意味しています。この意味ではX-BASICそのものが特に難しいということはないはずなのです。

問題になるのは従来のBASICに対してどうであるかということでしょう。X-BASICには，いわゆるマイクロソフト系のBASICに比べて言語仕様で劣っている点はないはずです。とすれば，もっとも異なってくるのはプログラミングスタイルです。

ところで誰もがX-BASICを苦にしてきたかというとそうでもありません。ざっと見てX-BASICへの移行に苦労する人としない人の差は，BASIC以外になんらかの言語でプログラムを記述できるかどうかがひとつの指標になっているように思われます。これは従来のBASICのプログラミングスタイルが非常に独特なものであったことに起因します。

従来のBASICの偉大な点は，ほとんど「どう並べても動く」ということです。「プログラミングにはまず仕様を決めて，アルゴリズムを探し，それをコーディングする」といったお決まりの手順を経ることなくプログラムを作成していくことが可能です。

このように，従来のBASICによるプログラミングが命令を「並べる」「つなぐ」といった感覚であったのに対し，X-BASICでは「組み上げる」といった感じの強いものになっています。処理の流れを線的に追っていくことから，処理の塊を組み合わせることがより重視されるといってもいいでしょう。

「どう並べても動く」ことが持つとっつきのよさはBASICのプログラミングを容易にしています。その半面，安易なプログラムがはびこる温床ともなっていたことも否めません。つぎはぎしていくうちに書いた本人にもわからなくなったプログラムや流れの読みにくいプログラムがBASICの名をおとしめていました。

X-BASICは旧世代のBASICよりはまともなプログラミング言語です。従来のBASICと同様な感覚で操作することもできますが，多少は型にはまったプログラミングスタイルを取ることを暗に要求しています。私の見るところではそれほど窮屈な制約ではありません。それは「プログラムを1行ずつ作るのではなく，ひとつの塊ごとに作るようにしなさい」というスタイルです。

ではここでいう「塊」とはなんでしょうか? それはすなわち関数に相当します。

関数の構造

X-BASICを学ぶとは関数を学ぶことです。「関数」とは数学に出てきたあの関数と基本的に同じものです。関数は写像とも呼ばれ，その定義は「ある集合Xの各要素に対して集合Yのなかの1個の要素を特定させる関係」とでもいえばいいのでしょうか。端的にいえば，

$$y=f(x)$$

これに尽きます。f()の部分を「ブラックボックス」，なにかものを入れたらなにか決まったものが出てくる「箱」として習った人も多いと思います。

このf()の部分が関数にあたります。数学の世界では数式で表されることが多いのですが，この部分はまさにブラックボックス，入力に対してなにかひとつの答えを出すものならなんでもかまいません。そのための法則がすなわち関数の実体です。

X-BASICで関数を使用するためにはプログラム中のどこかで処理内容を定義しておくことが必要です。

```
    func aaa(param)
    int i,j
         ⋮
         ⋮
    return(j)
    endfunc
```

のように“func〜endfunc”で囲まれた部分が関数の内容に相当します。ここではaaaが関数の名前，paramが引数になります。引数（ひきすう）とは，関数の入力に相当します。return()に指定されている j が関数の出力になります。入力に対する出力，これらが揃って数学的な関数は成立します。これが関数の基本形です。

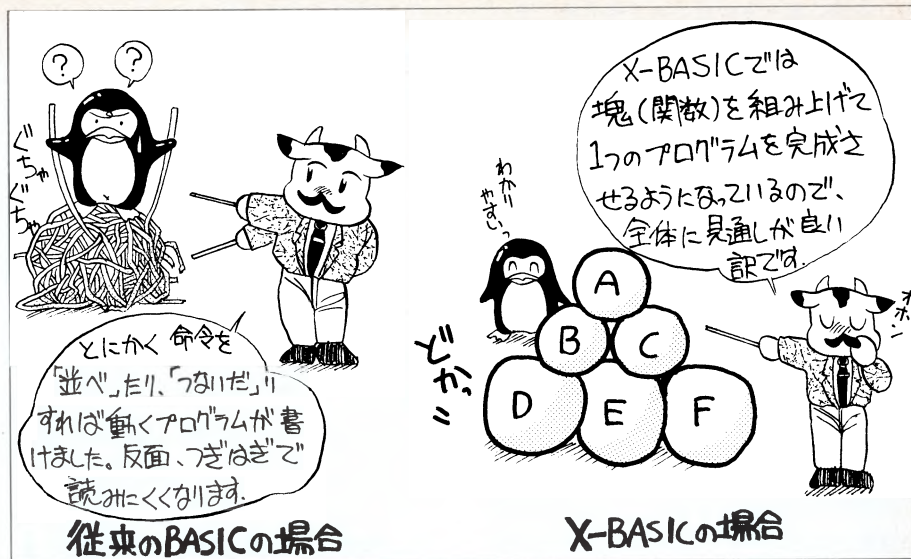
関数は値を返しますから演算時の部品として扱うことができます。

a＝b(c(x)＋d(e(f(y)＊16)＋x))

のように演算中に組み込むこともできますし，どこかで定義した関数を使って関数定義をすることもできます。こうして単純な関数を組み合わせて複雑な関数を作っていくことができるのです。

こう見ると関数というものもそう難しいものではないことがわかるでしょう。

先ほどの例では関数の内部で変数を宣言しています。この部分で定義される変数はローカル変数といい，この関数の内部だけで有効になります。

ローカル変数は関数が呼ばれるたびに新しく作られます。以前に使用していた値を使いたいといったときには関数の外で宣言した変数（グローバル変数）を使わなくてはなりません。逆に関数の外からローカル変数を参照することはできません。ローカル変数はその関数を実行しているあいだだけ存在するものだからです。

たとえば，

```
 10 int i
 20 a()
 30 end
 40 func a()
 50 int i=5
 60 b()
 70 endfunc()
 80 func b()
 90 print i
100 endfunc
```

というプログラムを実行した結果を予想してみてください。

答えは0です。関数a()はローカル変数 i に5を代入しています。そして，そのままb()を呼び出し変数 i を表示します。ここで参照される i はプログラムの先頭で定義されているグローバル変数の i です。プログラム中ではなんの値も指定されていませんが，グローバル変数は宣言されると同時に0で初期化されるので（文字列変数ならヌル文字），結果は0になるのです。途中の関数a()で宣言されていた i は関数b()を実行しているときには隠されてしまいます。X-BASICやC言語では同時に実行されているとみなされる関数はひとつだけですので，関数が入れ子になっていたとしても有効になるローカル変数はそのとき実行されている関数内で定義されているものだけとなります。

こういうややこしい話をするのは，言語によってはこれと違う考え方が必要なものがあるからです。PASCALなどではその関数を呼び出した親関数の変数も参照することができます（直系親族に限る）。これだとグローバル変数も親変数の一種と考えられます。PASCALをやらない人には関係ない話ですが参考までに。

ちなみにローカル変数にはグローバル変数と同じ名前の変数を使うこともできます。これはまったく別の変数として扱われます。関数自体も独立していますので，よその関数と同じ名前の変数を作ってもかまいません。BASICは現れた変数をまずローカル変数だと思って探し，でなかったらグローバル変数だと思って処理します。グローバル変数でもなかったら「変数が定義されていません」というエラーを出します（若干例外はありますが……）。

応用：構造化とモジュール化

ここまで関数についていろいろ説明してきました。「要するにサブルーチンのちょっと変わったやつ」としかとらえていない人もいることでしょう。

単に，GOSUB〜RETURNの代わりに使っているだけでは進歩はありません。関数を生かして使ってやることにより，従来のBASICとは違ったプログラミング環境を得ることができます。

もちろん，従来のBASICでも構造化とかブロック化などを行うことは可能でした。また，X-BASICでもスパゲティなプログラムは作成できます。これはプログラマの裁量によるところが大きいのですが，それがより自然であるという意味でX-BASICのほうが綺麗なプログラムを作成しやすい

のは間違いありません（ある程度使っている人なら，X-BASICでGOTOを駆使したプログラムというのは非常に作成しにくいということをご存じでしょう）。

また，X-BASICはC言語に変換できることにより非常にポテンシャルの高い言語となっています。同人ソフトなどではアクションゲームやシューティングゲームが平然とBASICで開発されています。多少の制限を除けばC言語と同じ扱いでコンパイルされるのですから，効果は劇的です。その結果，X-BASICの場合，コンパイルすることを前提としてプログラムを作成するのが半ば当たり前になってきました。X-BASICの完成されたプログラムとはコンパイルされたものだと思っておいたほうがいいでしょう。

そしてC言語の従兄弟のような書式は自然にユーザーをC言語へ移行させる効果も持っています。そのような際にはさらに関数の持つ意味は拡大されてくるでしょう。

それでは関数によって獲得できるであろうメリットというのはなんでしょうか？それはたぶん，

構造化がしやすい

モジュール化ができるかもしれない

ということでしょう。構造化はいうまでもなく処理の流れを明確にし，作成したプログラムを読みやすくしてくれます。

モジュール化というのは大雑把にいって汎用の部品として使えるプログラムを作ることです。いまはあまり深く考えないでおきましょう。

ここではひとつのプログラミングのスタイルを提示します。これは構造化，モジュール化といった部分に焦点をあてたプログラミングスタイルです。場合によっては気にいらない考え方だと感じる人もいるでしょうし，なんでこんな窮屈なことをするんだと思う人もいるでしょう。それはそれでかまいません。本来プログラムは自由なものであり，各個人でそれぞれ自分のスタイルを持っていくのが本道です。特にBASICではスタイルがないのがスタイルのようなものでした。一方，PASCALなどになると，ほとんど誰が書いても同じ処理は同じコードにしかならないんじゃないかと思うくらい整然とした構造ができあがっています。

ここではあえてスタイルを限定して話を進めていきます。

*　　*　　*

関数はそれ自体が独立したプログラムだと考えてください。引数を受け取ってなんらかの値を返すプログラムです。そのあいだの処理をするものは一般的なBASICプログラムとなんら変わりはありません。問題はどのような方針で関数を作るか？という点にかかっています。無論，正しい方法などは決まっていませんが，以下にいくつかの指針を挙げてみましょう。

●関数内でグローバル変数を使用しない

これは関数をモジュール化するためにも有効ですし，プログラムの構造を明確にする意味でも効果を発揮します。

必要なデータはすべて引数で受け取り，内部で完結したプログラムを書き，メインプログラムに値を返します。

このようにして作成された関数はいろいろなプログラムで使い回すことができます。変数の衝突が発生しないので，関数名さえぶつからなければ一度作ったものが無駄になることはありません。

こういったアプローチの持つ欠点はプログラムの記述が面倒になること，実行速度が多少落ちることなどです。また，X-BASICではローカル変数だけで記述するとプログラミングの効率が悪くなったり（特に大きなデータを扱うとき），記述しきれなかったりすることもあります。

関数では呼び出されるごとに変数を確保しますし，値の受け渡しを行いますので多少の効率低下が考えられます。しかし，コンパイルを考えた場合，GCCなどではローカル変数は優先的にレジスタに割り付けられますので（しかしver.1.Xではグローバル変数はレジスタに割り付けられません），ローカル変数を多用したほうが有利になることもあるのかもしれませんが……。

●関数を単機能化する

関数はたくさんの引数を持つことができますが，返り値として指定できるのはひとつの値だけです。これでは関数のなかで複数の処理をまとめて行うことは不自然になります。特にグローバル変数をアクセスしないプログラムでは頼りになるのは返り値だけですから細分化しないと十分な情報が得られないこともあるでしょう。

単機能化された関数は，関数の内容もわかりやすく，プログラムの可読性も向上することが期待できます。

半面，関数が単機能化されていくということは関数呼び出しの回数が多くなっていくということも意味しています。関数呼び出しの効率の悪さは，速度優先のプログラムにとってはあまりありがたいことではありません。

また，X-BASICではグローバル変数を使用しないという制約を加えると，まとまった単位で関数化しないとほとんどの処理をメインルーチンに押しつけることになってしまうという矛盾点を内包しています。

●引数，返り値を省略しない

先ほどから，入力に対して出力がひとつ決まるものを関数と呼んでいました。しか

ワンポイントテクニック

私がX-BASICでプログラムを組む際に真っ先に手を抜くのがユーザーインタフェイスです。定数はプログラムに埋め込んだり，inputを並べてすませてしまいます。コンパイル派の皆さんはたまにはC言語のようにコマンドラインからパラメータを受け取りたいと思うことはないでしょうか？

bc.xで変換したプログラムを見ていると，

```
b_argc
b_argv []
```

というものがみつかります。ちょっとC言語を知っている人ならこれがコマンドラインパラメータの受け渡しのために使えそうだと気づくでしょう。これをそのまま使ってやればいいのです。プログラムが実行されたときb_argcという変数（int）にはコマンドラインから渡されたパラメータ数＋1の値が入っています。ここでパラメータ数を調べ，パラメータが指定されているようならそれを受け取りましょう。最初のパラメータがファイル名なら，

```
filename＝b_argv(1)
```

となります。bc.xで変換されるので[]でなく()を使うのがポイントです。2つめのパラメータが数字なら，

```
num＝val(b_argv(2))
```

となります。ただし，これをインタプリタで実行すると変数未定義のエラーになりますので，少々邪道っぽいのですが，

```
c = 0 = 0
if c=1 then  {  ……  }
```

のようなインタプリタでは実行されない部分を作ってその中に入れておきましょう。ちなみに，“c＝0＝0”は0が0と等しいかどうかの論理値を変数cに代入しています。X-BASICとCでは論理値の真の値が異なりますので（1と－1），どちらで動いているか判別できます。

し，X-BASICではこれらはどちらも省略されることがあります。これでは関数といえないのですが，X-BASIC(というかC言語）ではこれも関数として扱います。このようなものは単なるサブルーチンとしての機能しかありません。多くのプログラミング言語で通常は関数と区別して「手続き」と呼ばれています。

X-BASICには「関数」と「手続き的関数」の2つがあります。普通の関数はいままで説明してきたもの，手続き的関数はそれから引数と返り値を抜いたものです。

グローバル変数にアクセスしない手続き的関数でモジュールを作った場合，そのモジュールはある一定の処理をまとめて行う以外のことはできないものになります。モジュールとして考えると，手続き的関数はほとんど無視してもいいでしょう。ちなみに，X-BASICの標準関数で引数も返り値もないものはwipe()くらいのものです。

通常の関数では，グローバル変数にアクセスしない場合はたいてい入力が必要ですので入力についてはたいした問題はないでしょう。しかし，関数の目的が主にその副作用にある場合（ラインを引いたり，音を出したり）には別に返す値がありません。このような場合はエラーが発生しているかどうかを返すのがよいようです。

メインルーチンではエラーをチェックすることでいっそう安全なプログラムにすることができます。開発中のバグも軽減できるでしょう。プロのプログラマならきっとこういう風に書かなければならないのでしょうね。

この方式の欠点はやはり記述が面倒なことと実行速度が落ちる，プログラムサイズが大きくなるということです。たとえば，ゲームなどでは「エラーは発生しない」という前提でプログラムを作成するほうが正道であるといえます。しかし，開発効率の点からいってどちらがよいかというのは微妙な問題です。最初は堅く作っておいて，エラーチェックはプログラムが動くようになったら少しずつ抜いていくというのがよいのでしょう。モジュール化を前提とするなら手堅く作っておいたほうが無難です。

●関数の入り口と出口を明確にする

BASICでのGOTO文が嫌われるのは，これを使うと処理の流れが分散してしまうからです。ひと頃から構造化プログラミングが推奨され，現在ではそれほどグチャグチャしたプログラムを書く人はいなくなりました。BASICに構造化制御命令が備えられたことも大きいでしょう。

その結果，非常に特殊な処理を除けば，あらゆるプログラムは必ずGOTO文なしで書くことができます。さらに関数を使えば容易にプログラムをブロック化することができるのはおわかりでしょう。プログラムを塊で作っていくという考え方も生まれてきます。さらにそのブロック内での処理の流れを明確にするための方策が「入り口ひとつに出口ひとつ」化です。

関数の場合，入り口はひとつしかないのでよいとしても，出口が複数あることはたまにあります。返り値が確定するたびにreturn()で抜けていては流れがつかみづらくなります。無論，これらはひとつにまとめることが可能です。

モジュール化の限界

実際にX-BASICの関数でモジュール化を行おうとすると，どうしても表現力の壁のようなものにぶつかります。

ローカル変数の限界と返り値がひとつしかないことの限界です。せめて返り値に配列が指定できれば……（数学的には邪道かな？)。X-BASIC自体にもまだまだ不備な点はあります。

こういった点でかなり進化したプログラム言語であるModulaIIなどでは，そのモジュールが使うグローバル変数を指定できるなどの改善がなされています。モジュールの都合で自由にグローバルにアクセスできるメモリが確保できるのです。

結局のところ，モジュール化では複数のプログラムをいかに有機的に結合させるかという部分に焦点が絞られます。なんとかして結合のための接点を増やすことはできないでしょうか。

たとえば，なんらかのマネージャのようなものを介してデータをやり取りすることが考えられます。多少「不自由」でもグローバルに使えるエリアがあると同系列のモジュールを実行させる際の効率がかなり上がることはまちがいないでしょう。参考例として，Z's-EXを見てみましょう。そこではグラフィックの退避エリア（裏画面）が設定されていますので，対応したプログラム（もちろん独立している）が気軽に作成できます。

このようにグローバル変数のうち，どうしても必要になるものについては，各モジュールで共通規約を作っておくことができます。あとは，そこにアクセスする関数さえ用意してやればもっと柔軟なプログラムが作成できるようになるでしょう。もちろんプログラムのモジュール化もいっそう意味を持つようになります。

また，こういった考え方はSX-WINDOWのような独立した並列処理間でのプログラミングにも応用できそうです。

外部関数の効用

さて，モジュール化というテーマで話を進めてきましたが，これまではX-BASICでモジュールを扱うことは現実的ではありませんでした。これまでに定義した関数を自在に使うには，それをすべてプログラム中に持っておかねばならないからです。これはちょっとたいへんなことです。

しかし，関数の仕様をちゃんとメモしておき，今月号の田村氏のコンバータで外部関数にでもしておけば，作成するプログラム中にモジュールライブラリを含む必要はなくなります。さらに，作成中のプログラムからモジュールを切り出して（それなりのデバッグは必要だが)，外部関数に分離すればずらずらと長いリストを扱うことから解放されます。もちろんデバッグ時の実行速度を稼ぐこともできます。外部関数によるモジュール化によってX-BASICの開発環境はどんどん充実させることができるでしょう。

いろいろな障害はありますが，ようやくモジュール化というものが多少なりとも現実味を帯びたものになってきたのです。モジュール化がもたらすメリットとデメリットをどう評価するかは各個人で違うでしょうが，X-BASICではこういったパラダイムによるプログラミングも不可能ではないのです。これもひとえにX-BASICが持つユーザー定義関数の柔軟性によるものです。X68000で「もっとも手軽な言語＝X-BASIC」の持つ可能性を見つめ直してみてください。

BASIC

多角形の最適基本図形分割
モーフィングへの第一歩

Shibata Atsushi 柴田 淳

昨今流行のグラフィック技法，モーフィング。これを実現するにはどうすればいいのか？　と柴田君は考えました。そして自己流でアルゴリズムを作りX-BASICで検討します。はたしてうまくいくのでしょうか。

かなり前になるが，テレビで古今東西の美人顔をすべて平均化したらどんな顔ができあがるか，という特番を見たのを記憶している。

まず，美しいとされている女性テレビタレントの顔と，海外の映画女優の顔。次に歴代のミスユニバースの顔，そして美人と伝えられる歴史上の人物の顔をワークステーションに取り込んで，のわわわわーと平均化していく。全部で100を超すかと思われる美人を，2，3分かけて連続的に合成していくその様はまさに圧巻で，いまだに僕の脳裏に焼きついている。

なお平均した美人がどんな顔になったかというと，これがなんとも当たりさわりのない，ごく普通の顔になったのだった。まあ屏風絵の小野小町とかが混ざっていたから，そのあたりにも問題があったのだろう。その番組が放映されたのは，モーフィングという言葉すら，まだ世間に知られていない時代のことであった。

モーフィングといわれて首を傾げる人でも「マイケルジャクソンのビデオで……」とか「ごっつええ感じの……」とか，通好みのところでは「スタートレックVで……」といえば思い当たるのではないか。つまり，人間の顔などの2つの画像の間を，連続的に変形させる技術をモーフィングという。前述の番組では，変形を中間でやめてしまうことによって2つの顔を合成していたのだ。

ちなみにmorph（モーフ）とは生物用語で「変態」，つまり昆虫が幼虫→サナギ→成虫と形態変化をみせるアレのことを指す言葉である。余談だが，するとイナズマンなどはヘンタイヒーローなわけだな。

ついひと昔前まではワークステーションクラスでしか実現しなかったこの技術も，最近ではMacintoshとかAMIGAで走るモーフィングソフトが出回っているらしい。だから当然，我らがX68000でもできないかということになるのだが，問題がいくつかある。

モーフィングの基本原理

なによりの問題は「どうすればできるか」ということ。モーフィングに関する解説書でもあれば問題は解決するのだけど，少なくとも僕はそんなの見たこともない（第一そんな本出したって売れなそうだし）。だけど僕自身，モーフィングの方法について断片的に知っていることはあるので，そこらへんから全体の基本原理みたいなものを推測してみたい。

●断片的情報その1：オブジェクト（変形させる対象のこと）は人間側が指定しなければならないらしい。

これは考えると当たり前のことである。たとえばある人間の顔から，別の人間の顔に変形を行う場合を想定すると，片方の左目はもう一方の人間の左目へと変形していく。

人間が見れば画面のどの部分が左目で，変形先の画像のどの部分に向かって変形させればいいかを理解するのにさほど苦労はしないが，このような仕事をコンピュータにやらせようとすると，不可能ではないにしろかなりの無理がともなう。

そこで，変形前の画像の左目にあたる部分を，まず多角形で囲ってやる。そのあと，変形先も同様にして多角形で囲うのである。実際の作業ではすべての象徴的な部分（輪郭がはっきりしていたり，またはうまく変形させたいと思う部分）をこのように切り出さねばならない。

要するに，画像だけ取り込めばあとは全自動で，というわけにはいかないのだ。でも逆に考えると，変形元，変形先のオブジェクトを任意に設定できるようになっている，ということなのかもしれない。髪の長い女性から，スキンヘッドのオジヤンに向けて変形を行う場合などで，変形元の女性の髪の毛をオジイの頭のシミに変形させたり，あるいは眉毛に変形させたりといった選択が可能なのである。

●断片的情報その2：切り出した多角形は基本図形に分割するらしい。

モーフィングに似た技術に，メタモルフォーゼというのがある。範囲をCGに限定しないなら，このメタモルフォーゼはアニメーションの歴史が始まったときから存在する技術であるといえる。クシャクシャになった線がだんだんと伸びていって，人間になったり動物になったりするアニメは誰でも見たことがあるだろう。

このメタモルフォーゼをコンピュータのディスプレイ上で実現するのは，非常に簡単である。変形前と変形後の座標データを用意しておき，対応する点同士を直線補間したものを次々表示していくだけでいい。

そこで当然考えるのが，モーフィングでも2つの多角形の間を補間しなければならないのだから，このメタモルフォーゼの原理を使えばいいのではないか，ということだろう。だけど実際はそう簡単にはいかないのである。

モーフィングの目的は与えられた多角形同士を補間するのではなく，多角形内に含まれる画像同士を補間することなのだ。もちろん多角形の頂点座標は直線補間されるのだけど，問題は多角形に囲まれた範囲がどのように「歪む」かということで，補間された多角形の頂点情報だけから，この歪みを一義的に決めるのはちょっと無理がある。出力に比べて，入力される情報が散漫すぎるのである。

ここで，すべての多角形は基本図形に，もっと一般的な記述を使えば三角形に分割することができる，という幾何学の定理を思い出そう。そして頂点の多い多角形内の画像よりも，三角形内の画像のほうがはるかに扱いやすいということは容易に想像で

きるだろう。三角形同士であれば，なんとか画像の自由変形ができそうだ。

扱いにくい問題を扱いやすいように小規模な問題に分割して，それぞれを解決した総体を元の問題の解とする，という手法はいたるところで見受けられる。で，モーフィングのような問題においても，この方法論は生かされて当然である。

ではここで一応，モーフィングを実行するにあたって行われていることをひと通りまとめてみよう。

まず，変形させたいオブジェクトを多角形として切り出す。次に変形先のオブジェクトも同様にして切り出す。なお対応する多角形は頂点を同一数持つようにしておく。

切り出しが終わったら，変形元と変形先の多角形を，双方に矛盾のないように基本図形に分割していく。矛盾のないようにとはどういうことかというと，分割によって生じる多角形の対角線が，変形元では多角形の内部に完全に収まっているのに，変形先の多角形では外周と交差しているようなときは変形がうまくいかないわけで，このような場合を指して「矛盾のある分割」というのである。

そしていよいよ，画像を変形させる。10段階で元画像から目的の画像に変形させる場合を考えると，最初に元と先の多角形の対応する座標を直線補間して，その直線上の10分のnに相当する座標を求め，n番目の変形の頂点の座標とする。変形途中の多角形の頂点座標が決まったら，先ほど分割した三角形内の画像を，変形元，変形先の両方から変形途中の三角形の形に変形する。あとは2つの三角形内の色を補間して，変形途中の画像に張り付ける。

これで，基本的にはモーフィング（らしきもの）が実現するはずなのだ。しかし中野氏にお伺いをたてたところ，アレは三角形でなく四角形に分割するのではないかとか，だいたい三角形同士の自由変形などどうするのかとか，いくつか否定的な見解を承る結果となったのであった。でもさんざん論議したあげく，どうやら問題なさそうだということでいまこの原稿を書いているのだけど，一抹の不安を残しつつ，先を急ぐ僕なのであった。

最適な分割とはなにか

2つ目の問題。モーフィングに必要な操作のうち，どこまでを人間にやらせるかということ。

変形させたい画像の中で象徴的な部分は，人間側が多角形として切り出さねばならない，ということは前述した。これは操作者側からの最小限の入力操作であるが，ではその先で，コンピュータにとっていちばん難しそうな作業，多角形の三角形分割はどうか。ちなみに先のテレビ番組では，これをなんと人間がやっていたのだ。

しかし，これではあまりにも美しくないではないか。多角形の切り出しについてはまあしょうがないにしろ，三角形分割のような，単純作業の積み重ねでなんとかなりそうな問題くらいは（操作の簡素化というよりもヘビービューティを目指すといった意味合いで）機械にやらせてみよう，というのが今回の題目である。で，どうせやるのだから，いっそ最適な三角形分割を出力するようなプログラムを目指そう。

しかしながら，最適な分割とはなにをもって最適とするか。最小個の三角形で分割するのが最適だろうか？　いや待てよ，n角形は確かn－2個の三角形に分割されるはずだから，最小個の三角形分割など存在するはずがない。

ところで，モーフィングでは切り出された多角形を三角形に分割し，その三角形を切り張りして変形途中の画像を生成するわけだが，その場合に恐ろしいのが，三角形の継ぎ目が見えてしまうこと，つまり色の変化が急激で線に見えてしまったり，極端な場合はドットが黒く抜け落ちてしまうことなどだ。これを最小限に抑えるためには，分割する三角形の辺の長さの総計を最小に抑えればよい。

つまり，モーフィングを実行するにあたっての多角形の最適基本図形分割とは，三角形の辺の総和を最小にすることなのである。多角形は弦（対角線のことをここではこう呼ぶことにする）によって三角形に分割されるのだから，その弦の総和が最小，といい換えたほうがいいかもしれない。分割された三角形の辺の総和には，元の多角形の外周が含まれているからである。

フッフッフ。お待たせした。次はいよいよ，最適基本図形分割の方法とアルゴリズムの説明に入るのだ。

多項式アルゴリズムの達成

人間にとってはやさしい幾何学問題であっても，計算機で解こうとするとたいへん苦労する，という場合がある。あるいはなんとか計算機で解けても，非常に時間がかかるときなど。このような幾何学問題を専門に扱う学問領域に，計算幾何学というのがある。皆さんにお馴染みなところでは，クイックソートやヒープソートなどの高速なソーティングアルゴリズムも，実はこの研究分野の（わりと初期の）成果なのだ。

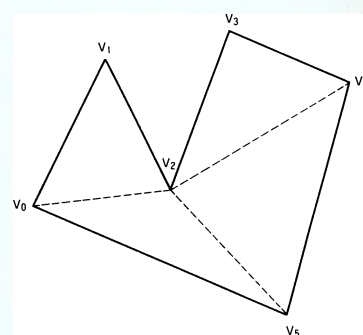
これから紹介するアルゴリズムも，ご多分にもれず計算幾何学の研究成果のひとつなのであるが，特筆すべきところは，なんといっても速度面において最適である，ということだ。

弦の総和の組み合わせをすべて求めて最適な分割を得ようとする。目的の多角形が凸である（へこんだり入り組んだりしていない）場合，可能な三角形分割の総数は，頂点の数が増えるにつれて指数関数的に増えていくのは十分予想できる。この指数関数的に処理時間が増加していくアルゴリズムというのは，プログラミングをしたことがある人ならわかると思うが，半ば絶望的といっていいほど効率の悪いプログラムなのだ。なんとかこれを，俗にいう多項式時間，つまり指数に変数がくっつかないような実行時間に抑えたい。

さてここで，問題を明確にするために，少々状況説明を加えよう。まず例に取る多角形をPと呼ぶ（図1を参照）。多角形の頂点には時計回りにv_0，v_1……と番号がふられており，連結する点同士は順番で見ても隣りあわせに並んでいるとする。ちなみにこのようなデータ構造を連結リストと呼ぶ。

次に部分多角形という少々耳慣れない概念を導入する。たとえば図1でv_2からv_5までの部分を$P_{2,4}$の部分多角形と呼ぶ。言葉で表せば，これは「点2を含めて数えて4番目の点までで構成される多角形」となるだろうか。ちなみに$P_{5,4}$といえば，v_5からv_2までの部分のことをいっていることになる。

なお，$P_{2,4}$には暗黙に弦(v_2,v_5)が含まれていることになる（閉じていないと多角形とはいえない）が，この弦の長さを$w(i,k)$というように定義しておこう。したがって，

図1

この多角形における最適な三角形分割

部分多角形$P_{2,4}$の外周の長さにはw(2,5)が含まれているといえる。以上の点を踏まえて，最適な三角形分割を求めるアルゴリズムを得るところまで話を進める。

さて，話は多少戻る形になるが，すべての可能な三角形分割を求め，そのなかから弦の長さの総和が最小になるような分割を求めようとすると，絶望的な時間がかかると前述した。で，その場合なにが問題なのかというと，これは明らかに，「分割のしかたになにも制限を加えていない」ことが問題なのである。もっといえば，「最適分割を出力することを妨げず，かつ幾何学的に理に適った分割方法の制限」が見つかりさえすればいい。

せっかく導入した部分多角形という概念をここで使うことになる。元の多角形Pを弦の総和が最小になるような部分多角形に分割していき，その作業を部分ごとに，三角形になるまで再帰的に繰り返せば，最適解が得られるはずである。しかし異なる部分多角形の種類はまだ指数関数のオーダーで存在するので，これでは問題はちっとも進展していない。

この泥沼の状況を著しく進展させるのは，次のような幾何学的性質である。耳の穴かっぽじってよーく聞くのだ。

「最適な部分多角形分割のなかには，それ自体すでに三角形になっているものも必ず存在する」

これはどういうことかというと，まず図1に点線で書き込まれたPの最適解を見てもらうとわかるのだが，たとえば部分多角形$P_{0,3}$は，最適な部分多角形分割でありながら，おまけに三角形でもある。このように都合のいい分割が見つかると，$P_{0,3}$はハサミでチョン切ったようにもう考えに含めなくてもよくなる。

そうやってどんどん，都合のいい三角形をハサミで切っていって，最後にひとつの三角形が残るまでその作業を繰り返すことによって，最適な三角形分割を得ることができる。

では，実際にどのようにすればこの都合のいい三角形が見つかるか。n角形のすべてのひとつ飛ばしで隣りあった対角線の長を求めて，少ないものから順につないでいけばいいと考えるかもしれないが，それでは線が交差する場合を避けられない。それにひとつ飛ばしで隣りあった頂点しか結ばないのであれば，点の多い多角形では真ん中がぽっかり残ってしまう。

ここでもう一度，先ほどの部分多角形に登場してもらおう。ある部分多角形$P_{i,t}$において，最適な三角形分割の長さを考える。このとき，2からt－1まで変化する変数kを想定し，図2のように，v_iからv_{i+t-1}までの点の並びを走査するとしよう。

また，部分多角形$P_{i,t}$に対する最適三角形分割の弦の総和を$\tau_{i,t}$と定義する。走査中の3点（v_i,v_{i+k-1},v_{i+t-1}）は三角形を成すことは明らかだから，$P_{i,t}$における最適な三角形分割の長さは，

$$w(v_i,v_{i+k-1})+w(v_{i+k-1},v_{i+t-1})$$

に，$P_{i,k}$と$P_{i+k-1,t-k+1}$における最適分割の弦の総和を足したものとなる。

ただし，k＝2のときは（v_i,v_{i+1},v_{i+t-1}）の三角形と，$P_{i+1,t-1}$の部分三角形の2つの部分にしか分割されない。同様にk＝t－1のときは，三角形（v_i,v_{i+t-2},v_{i+t-1}）と$P_{i,t-1}$の2分割に終わる。

これらのことを考えあわせると，式1に示すような$P_{i,t}$に対する最適分割を求める漸化式が得られる。ちなみにminとは最小元と呼ばれる操作で，集合のなかから最小値を求める操作を表している。

この式こそ，多角形分割がうまく三角形になる場合を考慮しつつ，頂点の多い多角形もちゃんと埋めてくれる分割のカギとなる計算方法なのだが，ここでまた新たな問題が生まれる。

さいわいX-BASICは再帰処理をサポートしており，したがって式1のような漸化式（式の内部に自分自身が入れ子になっている式のこと）を計算するのに便利なのだが，漸化式の中の$\tau_{i,t}$をその都度計算し直すとすると，これまた指数関数的に時間がかかってしまうのである。入れ子の計算中にもまた入れ子が含まれているという，もう本当に破綻的な状況がある。

ああ，せっかく指数関数時間のアルゴリズムがひとつ消散したと思ったのに，間髪入れずこのザマだ。さあ，どうする俺よ！

図2

このようにkを時計回りに走査していく

式1
漸化式

$$\tau_{i,t}=\min\{w(i+1,i+t-1)+\tau_{i+1,t-1},$$
$$w(i,i+t-2)+\tau_{i,t-1},$$
$$\min_{k=2,\ldots,t-1}\{(w(i,i+k-1)+$$
$$w(i+k-1,i+t-1)+\tau_{i,k}+\tau_{i+k-1,t-k+1})\}\}$$

動的計画法による問題の解決

実はここからが計算幾何学の本領発揮なのである。手っ取り早くいうと，$\tau_{i,t}$の計算には多項式時間あれば十分なのだ。この動的計画法に基づいたアルゴリズムは，劇的といっていいほど威力を発揮する。

さて，先の漸化式を解くとき，tを4から多角形の頂点数までまず増やしていくことにし，これを外側のループとする。なぜtが4からなのかというと，3以下の場合は部分多角形が三角形になってしまい，図2にあるような操作ができないからである。だからt＝1，2，3のとき，$\tau_{i,t}$の値は常に0となる。

次に内側のループとして，iを0から多角形の頂点数－1まで増やして$\tau_{i,t}$を計算していく。なんださっきと変わらないじゃないかと思うかもしれないが，ところがどっこい，漸化式中のkは常にtより小さく，またkはいつも正の値を取ることを考えるとt−k+1<tであることも明らかだ。これはどういうことかというと，$\tau_{i,t}$の計算に必要な，$\tau_{i+1,t-1}$と$\tau_{i,t-1}$，それに$\tau_{i,k}$，$\tau_{i+k-1,t-k+1}$の値は，すべて計算済みであるか，0であるということなのだ。

つまり，$\tau_{i,t}$の値を，下から順序よく積み上げていくことによって，重複なく計算していき，総計算時間を多項式時間に抑えることができるのである。実際BASICでプログラムを組んでも，それほど遅いと思わないようなものができあがるはずである。

さて，ここまでで計算速度上の問題点はすべて消え去った。あとに残された問題は，このアルゴリズムをどうやってコードに落とすかということ。ついでにいままで書かなかった，細かい最適三角形分割の方法にも触れていこう。

いよいよプログラミング

まず必要なのが，多角形を入力するインタフェイスとなる部分である。これはできの悪いお絵描きソフト程度のものなので，あまり特筆に値しないから省略。

与えられた多角形の連結リストを，便利のため時計回りに並べ替える。連結リストは当然配列にストアされているわけだが，この並びが時計回りかどうかを調べるには，図形の符号付き面積というのを調べればい

い。6400行からのarea()という関数がその作業をしているのだが，つまりは符号付き面積が，プラスならば反時計回り，マイナスならば時計回りなのである。ただし画面の座標系では縦軸の正負方向が幾何学の座標系とは逆なので注意が必要だ。

並べ替えが終わったら，次に$\tau_{i,t}$を求めるための下準備として，w(i,t)，つまり弦の長さを求めておく。5500行からのwcalc()という関数がそれにあたる。

ここで気をつけなければならないのは，弦が多角形の内部に完全に含まれない場合。このような点は分割のときに使ってもらっては困るので，無限大と思われるような大きな値をブチ込んでおく。そうすれば，漸化式中にw(i,t)が現れたとき，その項はとんでもなく大きい数字になるわけで，当然最小値を選ぶときに省かれることとなる。

なお，弦が多角形の内部に含まれる条件というのは，まず弦を結ぶ2点を挟む角の中に完全に含まれていて，なおかつ多角形の外周と交わらないことである。その判定は関数incl()で行われている。

ここまでの下準備が終わったら，いよいよ$\tau_{i,t}$の値を求める。ただし$\tau_{i,t}$のうちt>3のものには−1が代入されている。関数tau()において，$\tau_{i,t}$の値，つまり配列t(i,t)を調べて−1でなかったら値は決定しているのでその値を返し，そうでなければ新たに計算するという手順を踏む。計算に必要な入れ子のτはすべて前もって決まっていることは先に述べたとおりだ。

ただしここで注意しなければならないのは，$P_{i,t}$が多角形を成さない場合，具体的には弦（v_i,v_{i+t-1}）が多角形内に含まれない場合である。このときは$\tau_{i,t}$に無限大と見なされるような大きな数値を代入して，分割線として選ばれないようにしなければならない。図1の多角形を最適に分割したときに算出されるw(i,t)と$\tau_{i,t}$の値を表として載せておくので，参考にするといいかもしれない。

このようにして求められたすべてのτを元に，ついに最適な三角形分割を導き出すわけである，が，その前に，τの性質について少々考えてみよう。

もう一度だけいうと，$\tau_{i,t}$というのは部分多角形$P_{i,t}$における最適な，つまりは最小となる三角形分割の弦の総和を選び出すようになっている。では，このτのなかでさらに最小のものとはどんなものだろうか。

まず，式1の漸化式のうち前半の2つの値が断然有利であるのはわかっていただけるだろう。だいいち足し合う項が少ないのが目につく。

そのなかでさらに，tの値が少ないものがこれまた有利である。たとえばtの値が4である場合などは，1本の弦の値だけを持つものが最適な弦の総和として選び出される。要するにτのなかで最小になるのは，必ずt=4のなかから現れるのである。で，その分割というのは，弦が1本という部分からすでに予想がつくと思うが，最適分割の三角形として切り落とされるべき三角形なのだ。

では，tの値が増えていくと，τの値は漸化式の集合のどの部分から選ばれるだろうか。今度は逆に，前の2つの値は不利になる。まず弦の長さwの値が無限大になることが多くなる。だからマメに多角形の広い部分を走査している，後半の集合（漸化式の2番目のmin以下の部分）にとって断然，分がよくなる。

多角形を最適に，しかも真ん中が抜けるようなことなく分割できる秘密はまさにτのこのような性質に潜んでいる。で，この

表1

w(i,t)の表

	0	1	2	3	4	5
0：	0.0	∞	86.7	∞	∞	∞
1：	∞	0.0	∞	∞	∞	∞
2：	86.7	∞	0.0	∞	128.8	90.1
3：	∞	∞	∞	0.0	∞	151.1
4：	∞	∞	128.8	∞	0.0	∞
5：	∞	∞	90.1	151.1	∞	0.0

$\tau_{i,t}$の表

	4	5	6
0：	∞	∞	∞
1：	∞	∞	∞
2：	128.81	218.95	∞
3：	∞	∞	∞
4：	∞	176.82	∞
5：	86.68	176.82	∞

リスト

```
1000 /* 多角形の基本図形分割
1010 /*
1020 /* JAN. 19th 1993  by (ats)
1030 /*
1040 dim px(50),py(50),conn(1,50),f(50)
1050 dim float t(50,50),w(50,50)
1060 dim float m(50)
1070 int max = 0,mx,my,maxc = 0,t1,t2
1080 float inf = 10000
1090  initialize()
1100 print"ファイルを読み込みますか？"
1110 if inkey$<>"y" then cut() else fin()
1120 draw( max,2,2 )
1130 ptrev()
1140 wcalc() : print "すべての弦の距離を計算し終わりました。"
1150 tauclr()
1160 taucalc()
1170 print "これから分割を行ないます。"
1180 divide()
1190 print"ファイルを書き出しますか？"
1195 print
1196 print
1197 print
1200 if inkey$="y" then fout()
1210 end
2000 func divide()
2010  /* ** 多角形を最適に分割する **
2020 int a,b,a2,b2
2030 float tm = 0,m1,m2
2040  repeat
2050   tm = taumin( tm )
2060   m1 = wire( t1,nxf(nxf(t1)) )
2070   m2 = wire( t1+t2-1,nxb(nxb(t1+t2-1)) )
2080   if m1 < m2 then {
2090     a2 = t1 : b2 = nxf(nxf(t1)) } else {
2100     b2 = (t1+t2-1) mod max : a2 = nxb(nxb(t1+t2-1)) }
2110   conn( 0,maxc ) = a2 : conn( 1,maxc ) = b2
2120   maxc = maxc + 1
2130   line(px(a2),py(a2),px(b2),py(b2),4)
2140    a = nxf(a2)
2150    f(a) = 1 :circle(px(a),py(a),5,1)
2160    tref( a )
2170  until maxc = max - 3
2180 endfunc
2500 func nxf( n )
2510 int i,j
2520  for i = 1 to max - 1
2530   if f( (n+i) mod max ) = 0 then {
2540     j = (n+i) mod max : break }
2550  next
2560  return( j )
2570 endfunc
2600 func nxb( n )
2610 int i,j
2620  for i = 1 to max - 1
2630   if f( (n-i+max) mod max ) = 0 then {
2640     j = (n-i+max) mod max : break }
2650  next
```

ような性質を生かしつつ，多角形を基本図形に分割していくためには，以下のような操作が必要である。

まず，τのなかで最小のものを見つける。また，その最小のものというのは必ずひとつの三角形を最適に切り出すようになっている。図1でいうと$\tau_{5,4}$が最小なのだが，この場合，部分多角形$P_{5,4}$から距離最小の弦を見つけ出し，つまり（v_0,v_2）を結ぶ。すると点v_1はもう関係ないのだから，頂点数分用意してあるフラグの配列の添え字1に1を代入する。

また，使い終わった$\tau_{5,4}$には当然大きな数値を代入しておく。さらに，切り落とされた点v_1に向かって線を引くような連結を持つτの値も大きくしておく。こうすれば，分割線が交差することはなくなる。

2回目以降は，最小距離を持つ弦を探索するとき，切り捨てられた頂点を表すフラグを考慮に入れる点が異なるだけで，基本的にはいまの作業を全頂点数−3回繰り返すだけで，最適な三角形分割に用いるすべての弦を出力することができる。

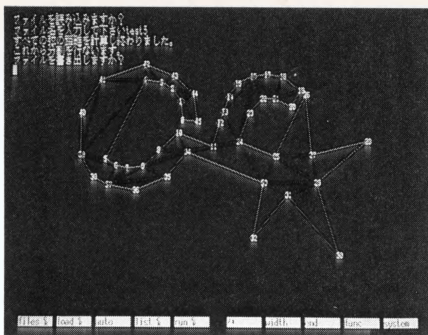
結構頑張るかわいいヤツ

このプログラムは，実をいえば速度的には最適ではない。まず検索などの小枝のアルゴリズムに改善の余地があるし，またBASICのプログラムであることを考えると，このようにさかんに関数を呼び出していては効率が悪い。

とはいえ10数個の頂点数の多角形であれば，1分と待たされることなく答えを出力するはずだ。そしてCに変換することによって，劇的に高速化すると予測されるのもまた事実である。このプログラムでは50頂点までの多角形を扱えるのだが，その最高頂点数の図形を入力しても，数分のオーダーで答えが出るだろう。

ここでプログラムの操作方法などを，必要ないと思うが一応書いておく。素直に入力して走らせると，まずファイルを入力するかどうか聞いてくる。これは誤入力を直すためのものだと思っていただければいい。だからたいていはここでY以外のキーを押すこととなる。

すると画面が変わって，図形入力の画面となる。画面左上に説明があるとおり，左クリックで頂点を指定して，間違ったと思ったら右クリックで消去する。このようにして多角形の外周が交差しないように（外周の交差判定はしていないのだ）点を置いていき，閉じないで両方のボタンをクリッ

```
2660  return( j )
2670 endfunc
2700 func tref( a )
2710 int i,j
2720  for i = 0 to max - 1
2730   for j = 0 to max - 1
2740     if i = a or (i+j-1) mod max = a then {
2750      t(i,j) = t(i,j) + 100000 }
2760   next
2770  next
2780 endfunc
3000 func taucalc()
3010  /* ** τを動的計画方(!)に基づいて計算する **
3020 int i,j
3030  for j = 4 to max
3040   for i = 0 to max - 1
3050    t(i,j) = tau(i,j)
3060   next
3070  next
3080 endfunc
4000 func float tau( i,j )
4010  /* ** 本文中の漸化式を再帰的に求める **
4020  /*(おおっ、スゴイぞ！と自分で自分を励ます俺)
4030 int k
4040  i = (i+max) mod max : j = (j+max) mod max
4050  if wire(i,i+j-1) >= inf then {
4060    return( inf ) }
4070  if t(i,j) <> -1 then {
4080    return( t(i,j) ) }      /*値が計算済みである場合
4090  mcl()
4100  m(0) = wire(i+1,i+j-1)+tau(i+1,j-1)
4110  m(1) = wire(i,i+j-2)+tau(i,j-1)
4120  for k = 2 to j - 1
4130   m(k) = wire(i,i+k-1)+wire(i+k-1,i+j-1)
4140   m(k) = m(k) + tau(i,k)+tau(i+k-1,j-k+1)
4150  next
4160  return( min(k) )
4170 endfunc
4500 func float taumin( tm;float )
4510  /* ** 配列 t(i,j) の中から最小値を見つけて返す **
4520 int i,j
4530 float mi = 10000
4540  for i = 0 to max - 1
4550   for j = 4 to max - 1
4560    if tau(i,j) < mi and tau(i,j) > tm then {
4570      mi = tau( i,j )
4580      t1 = i : t2 = j }
4590   next
4600  next
4610  t(t1,t2) = t(t1,t2) + 90000
4620  return( mi )
4630 endfunc
4700 func float min( n )
4710  /* ** m(n)の中から最小値を見つけ戻り値とする **
4720 int i
4730 float minimam = 100000
4740  for i = 0 to n
4750   if minimam > m(i) then {
4760     minimam = m(i) }
4770  next
4780  return( minimam )
4790 endfunc
4800 func float wire( a,b )
4810  /* ちょっとアコギな関数かも。記述は綺麗になるけど。
4820  return( w( (a+max) mod max,(b+max) mod max ) )
4830 endfunc
4840 func tauclr()
4850 int i,j
4860  for i = 0 to 50
4870   for j = 4 to 50
4880    t(i,j) = -1
4890   next
4900  next
4910 endfunc
4950 func mcl()
4960 int i
4970  for i = 0 to 50
4980   m(i) = 99999
4990  next
5000 endfunc
5500 func wcalc()
5510  /* *** すべての対角線分の         ***
5520  /* ***   ユークリッド距離を求める ***
5530 int i,j,h,a,b,c
5540  for i = 0 to max - 1
5550   w(i,(i+1) mod max) = inf : w((i+1) mod max,i) = inf
5560   for j = i+2 to max - 1
5570    if incl(i,j) = 0 then {
5580      w(i,j) = sqr(pow(px(i)-px(j),2)+pow(py(i)-py(j),2))
5590    } else {
5600      w(i,j) = inf }
5610  /* 弦が多角形内に含まれないなら距離を大きく取る
5620   w(j,i) = w(i,j)
5630   next
5640  next
5650 endfunc
5700 func incl( i,j )
5710  /* ** 指定した弦が多角形内に含まれるか **
```

分割された多角形

クすると，最後に指定した点と最初に指定した点が結ばれ，三角形分割のための計算に入る。

で，最適を豪語するプログラムはとかくイジメたくなるのが世の常だが，コイツは相当に痛めつけてもへこたれない。特にC変換された方は試してほしいのだが，梵字のように入り組んだ多角形すら，卒なくしかも最適に分割する。

と，実はこんな風に調子にのってもいられないのである。というのは，これでモーフィングを実現するために，最低限多角形の三角形分割に関する問題がすべて解けたかというとそうではないからだ。

残されている問題というのは，変形元と，変形先の多角形を双方矛盾なく分割するためにはどうするかということだ。ひとついえるとしたら，弦の長さを示すw(i,t)を，両方の弦の長さの和として取ったらうまくいきそうだということ。

たとえば変形元では多角形の内部に含まれている辺が，変形先では多角形の外周と交差する場合などは，変形元のwは有限の値を持つのだが，変形先では無限大となる。2つを足すわけだから結局無限大として扱われ，したがって変形元でも，分割弦としては選ばれないはずである。

あと残された大きな問題が，三角形同士の自由変型をどうするかということだが，逆にいえば，これさえ解ければわがX68000でもモーフィングのようなものが実現するのである。

開発力としてのX-BASIC

どーしてこんな重たい仕事を，わざわざBASICにやらせるのか。最後にここら辺をすっきりさせてから終わりたいと思う。

まず第一が，今月はBASICの特集だから。これはかなり大きな理由である。

第二に，これはまったく個人的な理由なのだが，僕の高校以来の友人に，僕と同い年のくせに偉そうに結婚シヤガル人がいて，

```
5720  /* **        どうかを判定する         **
5730 int a,b,c,d = 0
5740    a = tsize(j-1,j,i)
5750    b = tsize(i,j,j+1)
5760    c = tsize(j-1,j,j+1)
5770    if c > 0 then {
5780  /* 「開き」が 180度未満の場合
5790      if a < 0 or b < 0 then d = -1
5800    } else {
5810  /* 「開き」が 180度未満の場合
5820      if a < 0 and b < 0 then d = -1
5830    }
5840    if d = 0 then {
5850      for h = 0 to max - 1
5860       a = tsize( i,j,h )*tsize(i,j,h+1)
5870       b = tsize(h,h+1,i)*tsize(h,h+1,j)
5880       if a < 0 and b < 0 then {
5890         d = -1 : break }
5900      next }
5910  return( d )
5920 endfunc
6000 func float tsize( a,b,c )
6010  /* ** 三角形の符号付き面積の符号を求める **
6020 float ans
6030  a = (a+max) mod max : b = (b+max) mod max
6040  c = (c+max) mod max
6050  ans = (px(a)-px(c))*(py(b)-py(c))
6060  ans = ans + (px(b)-px(c))*(py(c)-py(a))
6070  return( sgn(ans) )
6080 endfunc
6200 func ptrev()
6210  /* ** 座標の並びが反時計周りなら **
6220  /* **       時計周りにする       **
6230  int i,a,b
6240  if area( max ) <> 0 then {
6250    print "座標の並びが反時計周りなので反転します。"
6260    b = (max-1) / 2
6270    for i = 0 to b
6280     a = px(i) : px(i) = px(max-1-i) : px(max-1-i) = a
6290     a = py(i) : py(i) = py(max-1-i) : py(max-1-i) = a
6300    next : wipe() : draw( max,2,2 ) }
6310 endfunc
6400 func area( max )
6410  /* ***** 点が時計周りか確かめる ****
6420  /*(注)ｸﾞﾗﾌｨｯｸ座標系ではY軸が反転している
6430 int i,a = 0,t = 0
6440  for i = 1 to max - 2
6450   t = t + px(i)*( py(i+1) - py(i-1) )
6460  next
6470  t = t + px(0)*( py(1) - py(max-1) )
6480  t = t + px(max-1)*( py(0) - py(max-2) )
6490  if t > 0 then return( 0 ) else return( -1 )
6500 endfunc
7000 func cut()
7010  /* ***** 多角形を切り出す *****
7020 int a,b
7030  cls : wipe()
7040  locate 0,0
7050  print "左クリック:頂点を登録";
7060  print "右クリック:頂点を削除"
7070  print "両クリック:多角形切り出し終了"
7080  mouse( 4 ) : mouse( 1 )
7090  if msopr() = 1 then {
7100   px( 0 ) = mx : py( 0 ) = my
7110   max = 1 : mswait()
7120   box( mx-1,my-1,mx+1,my+1,1 ) }
7130  repeat
7140   a = msopr()
7150    if a = 1 then {
7160      px( max ) = mx : py( max ) = my
7170      box( mx-1,my-1,mx+1,my+1,1 )
7180      line( px(max-1),py(max-1),mx,my,2 )
7190      max = max + 1
7200      np() : mswait() }
7210    if a = 2 then {
7220      max = max - 1 : np()
7230      wipe() : draw( max,2,0 )
7240      mswait() }
7250  until a = 3
7260  mouse( 2 )
7270 endfunc
8000 func np()
8010  /* ***** 頂点数を表示する ******
8020  locate 0,2
8030  print using "頂点数 ###";max
8040 endfunc
8050 func msopr()
8060 int a,b1,b2
8070  repeat
8080   msstat( a,a,b1,b2 )
8090   a = 0
8100   if  b1 = b2  and  b1 <> 0 then {
8110     if max < 4 then {
8120       locate 0,2
8130       print "頂点は4つ以上指定して下さい"
8140       beep } else {
8150       a = 3 }
8160   } else {
```

その御祝儀代と，披露宴で「てんとうむしのサンバカトリオ」という余興をやろうかと思っているので，その衣装代であるとか，とにかく僕自身が結構物入りであるということ。その分余計に原稿料を稼がねばならない。

ここからが真剣な理由。プログラムを組もうとするとき，Cで書いてもX-BASICで書いても同じようなコードができあがりそうだなあ，と思えるときがある。たとえば今回のような問題がいい例だ。具体的には大量の文字列やファイル操作などを扱わないで済むプログラムがそうではないだろうか。

で，そのうちで速度的にBASICでもイラつかない程度に収まることが予想されるものの場合，これはやっぱりBASICの出番なのである。

なぜか？　理由は簡単。BASICのほうがヘラヘラした態度で，つまり気楽にプログラミングできるからである。エラーメッセージは丁寧だし，だいいち終了したあとでも，変数の値を保持していてくれるのがうれしい。実行後やエラーで止まったあと，ダイレクトに変数を見られるというのはとてもありがたいことだ。ソースコードデバッガというのはややこしくていけない。そこまでユーザーフレンドリーでありながらCに非常に似た記述ができるのだから，便利なことこの上ない。

結論をいうと，今回のようにうまい具合に動くかどうかが不安な部分がある場合，そこを一度BASICで組んでみて動作確認をし，正しく動けばC変換してメインプログラムに組み込むというのは案外賢い選択なのではないかと思うわけだ。急がば回れという諺を引きあいに出すまでもないだろう。

もしかしたら，僕がCに不慣れだからこういうことをいうのかもしれないが，僕としてはやはり，X-BASICは開発力として高位に位置づけられる環境であるといいたい。いくら遅いといったって，最適なアルゴリズムを与えてやれば，今回のように結構見られるプログラムも組めるのであるから。

BASICというと初心者の使うものと思われがちだけど，ことその前にXがつくと，これは案外捨てたものではないような気がするのだが。

参考文献

計算幾何学，浅野哲夫，朝倉書店


```
8170     if b1 <> 0 then {
8180       a = 1 }
8190     if b2 <> 0 then {
8200        if max = 1 then {
8210         locate 0,2
8220         print "この点ばかりは削除できません
8230       } else {
8240         a = 2 } }
8250   }
8260  until a <> 0
8270  mspos( mx,my )
8280  return( a )
8290 endfunc
8300 func mswait()
8310  int a,b,c,d,e
8320  msstat( a,a,b,c )
8330  repeat
8340   msstat( a,a,d,e )
8350  until d <> b  or  e <> c
8360 endfunc
8400 func draw( max,col,mode )
8410  /* ***** 多角形を表示する ******
8420  /* モード0：多角形を閉じずに頂点を強調
8430  /*      1：閉じて頂点を強調
8440  /*      2：閉じて頂点を強調し、番号を表示
8450  /*      3：閉じて頂点は強調せず
8460 int a,i
8470  for i = 0 to max - 2
8480   line( px(i),py(i),px(i+1),py(i+1),col )
8490  next
8500  if mode < 3 then {
8510    for i = 0 to max - 1
8520     box( px(i)-1,py(i)-1,px(i)+1,py(i)+1,1 )
8530    next }
8540  if mode = 2 then {
8550    for i = 0 to max - 1
8560     symbol( px(i)-3,py(i)-6,str$(i),1,1,0,3,0 )
8570    next }
8580  if mode <> 0 then {
8590   line( px(0),py(0),px(max-1),py(max-1),col ) }
8600 endfunc
9000 func initialize()
9010  screen 2,0,1,1
9020  wipe() : cls
9030  palet( 0,0 )
9040  palet( 1,rgb( 30,3,0 ) )
9050  palet( 2,rgb( 0,30,5 ) )
9060  palet( 3,rgb( 31,31,31 ) )
9070  palet( 4,rgb( 3,2,31 ) )
9080 endfunc
9090 func fout()
9100 char a
9110 str fn
9120  input"ファイル名を入力して下さい",fn
9130  fopen( fn+".poi","c" )
9140  a = max : fputc( a,0 )
9150  fwrite( px,max,0 )
9160  fwrite( py,max,0 )
9170  fcloseall()
9180 endfunc
9190 func fin()
9200 char a
9210 str fn
9220  input"ファイル名を入力して下さい",fn
9230  fopen( fn+".poi","rw" )
9240  a = fgetc( 0 ) : max = a
9250  fread( px,max,0 )
9260  fread( py,max,0 )
9270  fcloseall()
9280 endfunc
9500 func prput()
9510  int i,j,h
9520  print : print "   ";
9530  for i = 0 to max - 1
9540   print using"  ##   ";i;
9550  next : print
9560   for i = 0 to max - 1
9570    print using"##:";i;
9580    for j = 0 to max - 1
9590     if w(j,i) >= inf then {
9600       print "  ∞   "; } else {
9610       print using"###.#  ";w(j,i); }
9620    next : print
9630   next
9640  print "   ";
9650  for i = 4 to max
9660   print using"  ##   ";i;
9670  next : print
9680   for i = 0 to max - 1
9690    print using"##:";i;
9700    for j = 4 to max
9710     if t(i,j) >= inf then {
9720       print "  ∞   "; } else {
9730       print using"###.## ";t(i,j); }
9740    next : print
9750   next
9760 endfunc
```

モジュール化を意識した
変形用関数の作成

Nakano Shuichi 中野 修一

X-BASICによる関数の作り方を実践してみましょう。できるだけモジュールとしてまとまったかたちで作成していきます。題材は結果のわかりやすいグラフィック関係を集めてみました。

能書きばかりいってるのもなんなんで，実践編です。

柴田君が作っていたモーフィング用のアルゴリズムで質問を受けました。モーフィングはそのうち誰かがやるだろうとは思っていたのですが，それが文系の柴田君だったとは……。直感的に，三角形ごとに独立して線形補間したのでは不自然になるような気がします。しかし，図形指定の際のサンプル点はけっこうな量になりそうなので，もしかしたら，これでもそれらしく見えるのかもしれないし……。結局，「やってみなければなんともいえない」ということに落ち着きました。

三角形の分割は柴田君がとうに完成していますので急遽，三角形の変形ルーチンを作ってみました。あとは，ブレンド処理部分を作ってユーザーインタフェイス（けっこうやっかい）を加えれば少なくとも「モーフィングらしきもの」はできるはずなのですが……。

ということで，X-BASICを使い，先ほど述べたモジュール化も意識してプログラムを作ってみましょう。

三角形の変形

問題の自由変形ですが，グラフィックツールなどで四角形ができているのですから三角形でできないはずはありません。四角形の自由変形を見ると，変換元の四角形を辺に沿ってスキャンしてやり対応する辺からライン状に描画しています。元のほうが小さければいいのですが，そうでない場合，同じピクセル上に複数のピクセルが対応する可能性があります。このときZ'sSTAFFでは次々と上書きされますのであとに描いたものが優先，MATIERではすでに描いてあるところには描画しないので先に描いたものが優先されます。

三角形でライン状にスキャンするというアルゴリズムを使うと，ある頂点の周辺ではラインの密度が極端に高くなりますから，見た目に不自然な描画が行われる可能性があります。思うに，四角形には凸型と凹型の2種類があるためこのような方法で描画しているのでしょう。三角形ならばもっと普通の方法で間にあいそうです。

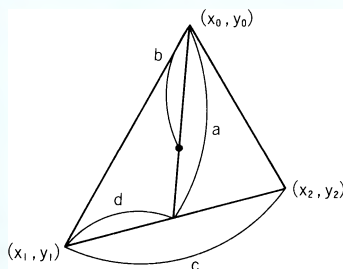
画像変形の基本は「変換後のピクセルから逆変換して変換前の色を探す」ことです。2つの三角形が与えられたとき，変換後のほうの三角形を構成する各ピクセルは変更前の三角形のどの部分から持ってこられたものかを順に求めていくわけです。

対応する点を求めるにはなんらかの座標系を与えてやらなければなりません。三角形の各頂点と三角形内部のある点との関係を示すものが必要なのです。これは図1のようにして対応づけます。

ちょっと複雑な部分（私にとっては）を取り上げてみましょう。

まず2点を通る直線の方程式が必要です。高校の頃さんざんやったような……。

いま考えると高校の数学の時間というのはいろいろ役に立つことを教えていたんですね。ベクトルの計算だとか，平面幾何とか，一次変換とか……。昔ちゃんと習ったはずなのに思い出せないから自分で式を導かなければならなくなります。あーあ。

図を描いてごちゃごちゃやって方程式を導きました。次に必要なのは2直線の交点

三角形の変形

座標です。これもさんざやった覚えはあるのですが……。なんとか解決してプログラム化します。まずは関数化してしまいましょう。リスト1です。

いきなり邪道ですが，関数は返り値をひとつしか持てないのでX座標とY座標を組み合わせて返り値としています。

この関数は，

kouten(x0,y0,x1,y1,x2,y2,x3,y3)

のように使用します。(x0,y0)－(x1,y1)を通る直線と(x2,y2)－(x3,y3)を通る直

リスト1

```
 10 input "直線2：始点";x2
 20 int x0,x1,x2,x3,y0,y1,y2,y3,a,b
 30 int err=-1
 40 input "直線1：始点";x0,y0
 50 input "直線1：終点";x1,y1
 60 input "直線2：始点";x2,y2
 70 input "直線2：終点";x3,y3
 80 a=kouten(x0,y0,x1,y1,x2,y2,x3,y3)
 90 if a<>err then {
100    print"交点=";a mod 512,a/512
110 } else print"画面内に交点はありません"
120 end
130 func kouten(x0,y0,x1,y1,x2,y2,x3,y3)
140 int x,y,err=&H7FFFFFFF
150 float a0,a1,b0,b1
160 a0=katamuki(x0,y0,x1,y1)
170 a1=katamuki(x2,y2,x3,y3)
180 b0=y0-a0*x0:b1=y2-a1*x2
190 if a1=a0 then {
200    x=err mod 512 :y=err/512
210 } else {
220      x=(b1-b0)/(a0-a1)+0.5#
230      y=x*a0+b0+0.5#
240 }
250 if x<0 or x>511 then x=-1:y=0
260 if y<0 or y>511 then x=-1:y=0
270 return(y*512+x)
280 endfunc
290 func float katamuki(x0,y0,x1,y1)
300 float a
310 if x0=x1 then {
320    a=(y1-y0+0#)/(x1-x0+0.5#)
330 } else a=(y1-y0+0#)/(x1-x0)
340 return(a)
350 endfunc
```

線の交点をまとめて返します。実際の座標に直すには,

x =kouten() mod 512

y =kouten()/512

とします。おわかりのように画面モードは横512ドットしか考慮していません。交点が画面上になかった場合はエラー(−1)が返ってきます。

点の指定のしかたによっては平行線ができる場合もあるので,エラーチェックをすることが必要です。範囲を制限しないとエラーを返すことが難しいのでこのような仕様になりました。

あとは対応する点を見つけてやればいいだけですので一気に片づけてしまいましょう。あと揃えなければならない処理は,

1) 2点で示された線分をn:mに内分する点の座標を求める関数

2) 2点で示された線分上の点がその線分をどの割合で分割するかを求める関数

3) 実際にスキャンを行う駆動部分

などです。

実際にコーディングしてみたところ,1)は関数化するまでもないとわかり,3)はメインルーチンのことです。

三角形の内部だけを処理するという部分でやや手間取りましたが,意外とすんなり動いてくれました。なお,X軸と平行な線と直線の交点を求める際に専用の関数を作りました。求めるのはX座標だけなのに返り値を分解してやらなければならないのが嫌だったからです。

作成したものがリスト2です。リスト1の関数を加えてプログラムを完成してください。

これを変形前と変形後の2つの三角形とのあいだに対して行って両者の色をブレンドするとモーフィングっぽいものができるはずです。ただし,今回は画面から画面への変換しか考慮していませんから,全画面にわたるモーフィングなどにはこのままでは対応できません。さらに,精度を上げたり,floatを使っている部分を固定小数点にして整数化するなどの改良は行ったほうがいいでしょう。

画像変換もう一発

どうも最近画像変形づいているような気がしますが,ついでにもうひとつ。Z's-EX以来,画像のフィルタが作りやすい環境というのが整えられているのにどうして誰も作んないんだーと思っていたのをやっつけます。

画像の基本的な変形として,カメラのレンズによる収差を真似たものを作ってみました。収差というのは画像の歪みのことで,たとえば四角いビルを写したときに胴のあたりが膨らむような場合を樽型収差,真ん中がへこんで見えるような場合を糸巻き収差といいます。

んで,四角形の真ん中を膨らますような変形というのはどのようにして実現するのでしょうか?

ビルではわかりにくいので,ここでアーチェリーの的のような同心円を考えます。これを収差の激しいレンズで見るとどうなるでしょうか。ビルの例に対応させてつらつらと考えると図2のような結果になるだろうと予想されます。

要するに中心部と周辺部で倍率が異なっているわけです。

理屈がわかればあとは簡単です。画面の中心からの距離に応じて画像を拡大/縮小させてやればいいのです。先ほどからいっているように,画像の変形は変更後の点がどこからきたものかを調べる作業です。ある点が画面の中心に対して拡大されている場合はちょっと中心よりの点を,縮小されている場合はちょっと外側の点を取ってくるようにするわけですね。

ここで全体の座標系を画面の中心を原点に取り直すとさらに考えやすくなります。

あとは画面をスキャンして中心からの距離を求め,距離に応じて座標に拡大率をかけるだけです。実際のプログラムでは,画面内には中心からの距離が同じ点が8カ所あるということを利用して処理を簡略化しています。

このような全画面を描き換えるプログラムでは元画面の情報をどこかに保存しておかなくてはなりません。関数がグローバル変数をアクセスしないという条件でこのプ

リスト2 三角形の変形

```
  10 int tx(5),ty(5)
  20 int a,b,dam,bl,i,j
  30 /*  じゅんび
  40 screen 1,3,1,1
  50.mouse(0):mouse(1)
  60 while 1
  70   pic_load("test.pic",0,0)
  80   for i=0 to 5
  90     repeat:msstat(dam,dam,bl,dam):until bl=-1
 100     mspos(a,b):tx(i)=a:ty(i)=b :pset(a,b,0)
 110     repeat:msstat(dam,dam,bl,dam):until bl=0
 120   next
 130   tmorph(tx(0),ty(0),tx(1),ty(1),tx(2),ty(2),tx(3),ty(3),t
x(4),ty(4),tx(5),ty(5))
 140   input i
 150 endwhile
 160 end
 170 /*  びょうが
 180 func tmorph(x0,y0,x1,y1,x2,y2,x3,y3,x4,y4,x5,y5)
 190 int minx,maxx,miny,maxy,x,y,xn0,xn1,xn2,w0,w1
 200 int kx0,kx1,ky0,ky1,a,c
 210 float p0,p1
 220 maxx=max(x3,x4,x5)
 230 minx=min(x3,x4,x5)
 240 maxy=max(y3,y4,y5)
 250 miny=min(y3,y4,y5)
 260 for y=miny to maxy
 270   xn0=ykouten(x3,y3,x4,y4,minx,maxx,y)
 280   xn1=ykouten(x3,y3,x5,y5,minx,maxx,y)
 290   xn2=ykouten(x4,y4,x5,y5,minx,maxx,y)
 300   w0=sqr(min(xn0*xn0,xn1*xn1,xn2*xn2))
 310   w1=max(xn0,xn1,xn2)
 320   for x=w0 to w1
 330     a=kouten(x3,y3,x,y,x4,y4,x5,y5)
 340     kx0=a mod 512
 350     ky0=a/512
 360     p0=p_length(x3,y3,x,y,kx0,ky0)
 370     p1=p_length(x4,y4,kx0,ky0,x5,y5)
 380     kx1=((x2-x1)*p1+x1)+0.5#
 390     ky1=((y2-y1)*p1+y1)+0.5#
 400     c=point((kx1-x0)*p0+x0,(ky1-y0)*p0+y0)
 410     pset(x,y,c)
 420   next
 430 if asc(inkey$(0))='q' then break
 440 next
 450 endfunc
 460 func max(a,b,c)
 470 int r
 480 if (a>=b) and (a>=c) then { r=a
 490 } else if b>=c then r=b else r=c
 500 return(r)
 510 endfunc
 520 func min(a,b,c)
 530 return(-max(-a,-b,-c))
 540 endfunc
 550 func ykouten(x0,y0,x1,y1,minx,maxx,y)
 560 int r
 570 float a
 580 if x0=x1 then { r=x0
 590 } else {
 600   a=(y1-y0+0#)/(x1-x0)
 610   if a=0 then { r=-10000
 620   } else {
 630    .r=(y-y1+a*x1)/a+0.5#
 640   }
 650 }
 660   if not (r>=minx and r<=maxx) then r=-10000
 670 return(r)
 680 endfunc
 690 func float p_length(x0,y0,x1,y1,x2,y2)
 700 float a0,a1
 710 a0=distance(x0,y0,x2,y2)+0.1#
 720 a1=distance(x0,y0,x1,y1)
 730 if (a1/a0)>1 then a0=1 else a0=a1/a0
 740 return(a0)
 750 endfunc
 760 func float distance(x0,y0,x1,y1)
 770 return(sqr((x0-x1)*(x0-x1)+(y0-y1)*(y0-y1))+0#)
 780 endfunc
 790 /* 以下はkouten(), katamuki()関数か加わるリスト1参照のこと
```

ログラムを作ると，ほとんどの処理はメインルーチンの負担になります。これではモジュール化の意味がありません。

ここではローカル変数だけを使う関数で実現した例と，グラフィックの待避部分だけグローバル変数をアクセスした例とを作ってみました。Z's-EXなどに移植する場合はグローバル変数参照版のほうがわかりやすいでしょう（実はこちらのほうが先に作られている)。

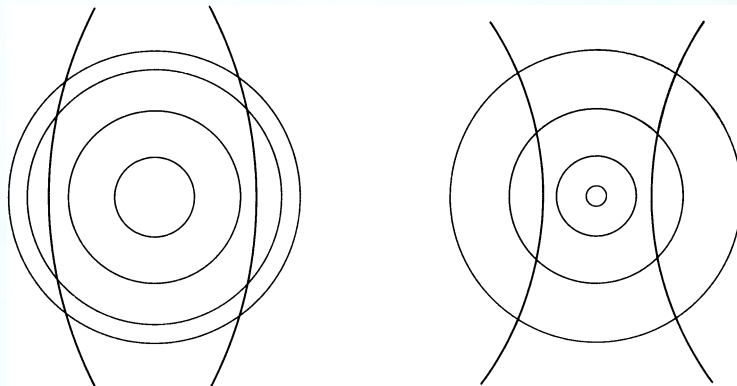
＊　＊　＊

いくつかの関数をモジュールとして作成してみたのですが，X-BASICではグローバル変数をどうしても使いたくなることがあります。デバッグの簡単さなどから，いつもはほとんどグローバル変数しか使わないのでいっそう強く感じるようです。

モジュール化については「継続は力」ですから，もっと蓄積ができなければ効果は表れないかもしれません。しかし変数の管理を徹底すると，できあがるプログラムのかたちが変わっていくのがよくわかります。無駄が取れた部分となにか無駄なのになあという部分がはっきりするのは，プログラムの流れが明確になっているからでしょう。これも効用のひとつかもしれません。

樽型収差の例

糸巻収差の例

図2

注：ちなみに収差の大きいレンズはあまりいいレンズではありません。

リスト3　収差変形(モジュール版)

```
 10 int g(511,511),err=-1
 20 int i,j,xy,x,y
 30 float k
 40 /*
 50 screen 1,3,1,1
 60 pic_load("test.pic",0,0)
 70 for j=0 to 511
 80   for i=0 to 511
 90     g(i,j)=point(i,j)
100   next
110 next
120 /*
130 repeat
140   input k
150   if k=0 then break
160   for j=0 to 255
170     for i=0 to 255
180       if j<=i then {
190         xy=shusa(i,j,k)
200         if xy<>err then {
210           x=xy mod 256
220           y=xy/256
230           pset(256+i,256+j,g(256+x,256+y))
240           pset(256-i,256+j,g(256-x,256+y))
250           pset(256+i,256-j,g(256+x,256-y))
260           pset(256-i,256-j,g(256-x,256-y))
270           pset(256+j,256+i,g(256+y,256+x))
280           pset(256-j,256+i,g(256-y,256+x))
290           pset(256+j,256-i,g(256+y,256-x))
300           pset(256-j,256-i,g(256-y,256-x))
310         }
320       }
330     next
340   next
350 until 0
360 end
370 /*
380 func shusa(i,j,k;float)
390 int x,y,r,err=-1
400 float d,f,z
410 if k>0 then {
420   z=1+1.41#/k
430 } else z=1+1#/k
440 /*
450 d=sqr(i*i+j*j)
460 f=(1+d/256#/k)/z
470 x=i*f+0.5#
480 y=j*f+0.5#
490 r=y*256+x
500 if x>255 or x<0 then r=err
510 if y>255 or y<0 then r=err
520 return(r)
530 endfunc
```

リスト4　収差変形(外部配列参照版)

```
 10 /*
 20 /*  レンズの収差を表現する
 30 /*
 40 /*  パラメータの符号  =  正  ：樽型収差
 50 /*                    =  負  ：糸巻き収差
 60 int g(511,511)
 70 float k
 80 /* 初期設定
 90 screen 1,3,1,1
100 pic_load("test.pic",0,0)
110 for j=0 to 511
120   for i=0 to 511
130     g(i,j)=point(i,j)
140   next
150 next
160 /* メインループ
170 repeat
180   input k
190   if k=0 then break
200   shusa(k)
210 until 0
220 end
230 /*
240 /*
250 func shusa(k;float)
260 int i,j,x,y
270 float d,f,z
280 if k>0 then {
290   z=1+1.41#/k
300 } else z=1+1#/k
310 /*
320 for j=0 to 255
330   for i=0 to 255
340     if j<=i then {
350       d=sqr(i*i+j*j)
360       f=(1+d/256#/k)/z
370       x=i*f+0.5#
380       y=j*f+0.5#
390       if x<=255 and x>=0 and y<=255 and y>=0 then {
400         pset(256+i,256+j,g(256+x,256+y))
410         pset(256-i,256+j,g(256-x,256+y))
420         pset(256+i,256-j,g(256+x,256-y))
430         pset(256-i,256-j,g(256-x;256-y))
440         pset(256+j,256+i,g(256+y,256+x))
450         pset(256-j,256+i,g(256-y,256+x))
460         pset(256+j,256-i,g(256+y,256-x))
470         pset(256-j,256-i,g(256-y,256-x))
480       }
490     }
500   next
510 next
520 endfunc
```

BASIC関数から外部関数を自動生成

BAS2FNC.X

Tamura Kent 田村 健人

X-BASICによるプログラムではないのですが，X-BASICで作成した関数を外部関数として使用するプログラムを発表します。モジュール化されたプログラムを外部関数化しておけば関数効率も向上することでしょう。

X-BASICの武器は，

1) 簡単にCに変換でき，高速実行が可能(BAStoC)

2) 自分で作った命令を追加できる（外部関数)

ですね。このうち1)のほうはほとんどなにも考えず誰でも恩恵を受けられるのに，2)となるとマシン語の知識を要求されます。マシン語でプログラムを組めない人たちは，X-BASICに用意されているこんなにおいしい機能を使えずにいるのです。これでは不公平ですね。そこで，BASICの関数を外部関数に自動変換するプログラムを作ってみました。

外部関数にする意義

自分で命令を作れるという点では,BASIC関数も外部関数も同じです。では，なぜ外部関数にすると嬉しいのでしょうか。それは，なんといっても実行速度でしょう。BASICはプログラムを解釈しながら厳密なエラーチェックもしながら実行します。それに比べ外部関数はプログラムそのものが動くだけです。余計なエラーチェックをする必要もありません。場合によっては数百倍も速くなります。

外部関数にすると，どのプログラムからでも使えることも忘れてはいけません。あるプログラムで使っていたBASIC関数をほかのプログラムで使いたいときには，目的の関数をわざわざsave@してからプログラムをロードして関数をload@しなければなりません。これは面倒です。外部関数ならいつでもどこでも使えるのです。

速度が必要ならコンパイルすりゃいいじゃないかという人もいるかもしれませんが，BASICの最大の利点はインタプリタ環境でトライ&エラーを繰り返せることです。そういったときにすでにできあがっている部分だけでも高速に実行できたら作業効率も向上します。このように作っていき，全部できあがったところでまとめてコンパイルすればいいのです。

問題は?

どうすればBASIC関数を外部関数にできるでしょうか。BASICのプログラムをマシン語にするのですから，もちろんBAStoCとCコンパイラを使いますね。しかし，当然ながらCコンパイラが作るマシン語はCの関数の形式であって，BASICの外部関数ではありません。では，Cの関数とBASICの外部関数の違いはなんでしょう。XCの「Cユーザーズマニュアル」と「プログラマーズマニュアル」にそれぞれの説明がありますので，比べてみましょう

●引数の渡し方

双方とも，スタックを利用しています。関数の引数にスタックを使うのはごく自然なことですね。Cの関数の場合,引数がdouble型では8バイト，それ以外の型では4バイトとして単純にスタックに積んであるだけです。これが外部関数では，引数がどの型でも，2バイトの型情報と8バイトの引数の内容にし，引数の総個数もつけます。

●返り値

Cでは,double型だとd0-d1レジスタ，それ以外ではd0レジスタに入れて関数を終了します。外部関数ではかなり大がかりで，d0レジスタがエラーフラグ，a0レジスタが返り値へのポインタ，a1レジスタがエラー時のメッセージへのポインタです。返り値は引数と同じように型情報（もどき）2バイトと内容8バイトにしてメモリ上に置きます。

*　　*　　*

C関数と外部関数の違いは，実はこれだけです。これぐらいなら簡単に変換できそうですね。

関数だけでは外部関数はできません。外部関数では「インフォメーションテーブル」というデータが必要です。インフォメーションテーブルの詳しい説明は「プログラマーズマニュアル」を見てもらうとして，ここで問題になりそうなのは「パラメータテーブル」です。各関数の引数と返り値を書く必要があります。

対処法と方針

引数の違いについては，実行時には引数の個数と型情報までついてくることですので，これを利用して「その場でスタックに引数を積み直すプログラム」を埋め込み，その中でBAStoCした関数を呼びます。

返り値は,「エラーは発生しない」と決めつけて,「返り値をメモリに置く」「そのアドレスをa0レジスタに入れる」「d0レジスタを0にする」というプログラムを付け加えます。

このように，レジスタを直接扱う必要があるので，引数と返り値についてはアセンブラソースに手を加えます。

パラメータテーブルを作るために各関数の引数と返り値を知りたいわけですが，アセンブラソースを見ても，これらの情報はすでに失われています。また，Cソースではどの行が関数定義なのかが非常に解析しにくそうなので,BASICのリストから解析することにします。

ところで，BAStoCするとCソース中に必ずmain関数ができます。main関数は外部関数に必要がないうえに，アセンブラソース中にCのスタートアップルーチンを必要とする命令を作ります。よって，削除しなければなりません。アセンブラソースの段階ではどこからどこまでがmain関数なのか判別できそうにないので，Cソースのときに削除します(XC専用ならアセンブラからでも削除できるが，GCCも使えるようにしたかった)。

なんと，アセンブラソース，BASIC，Cソース，「ABC」すべてのファイルに手をつけなければなりません。これには自分でも面食らってしまいました。

コーディング

ではプログラムの説明です。自動生成するためには子プロセスでBAStoCなどを呼び出さなければならないので，X-BASICで書くことはあっさりあきらめてCで書きました。さらにいえば，都合によりXC ver.1.Xには対応していません。GCCを使う場合もXC ver.2.0以降のシステム（ライブラリとBC.X）を用意しなければなりません（該当するところを変更してもよい）。

全体の流れは，

1)　BAStoCでCソースを作成

2)　Cソースからmain関数を削除(RemoveMain(　))

3)　CコンパイラでCソースをアセンブラソースにコンパイル

4)　BASICリストから各関数の返り値と引数の情報を得る(AnalyzeParameters(　))。

5)　インフォメーションテーブルをでっちあげ，引数と返り値を適合させるプログラムを埋め込む（MakeSource(　)）

6)　アセンブル，リンク

となります。

細かい説明にいきます。アセンブラの知識がないと理解しにくいかもしれません。

ソースの先頭あたりで定義している構造体ptablerecは関数の情報を記憶するためのものです。fnameには関数名を置きます。X-BASICの識別子の最大長の64文字分です。paramは引数と返り値の型です。X-BASICの引数の個数制限10個と返り値1個で11要素の配列です。外部関数で使う型定数と同じ値を入れます。任意個数の引数の型のあとに1個の返り値の型が続きます。paramnは引数の個数，stackは引数の型がfloatなら8，それ以外なら4として，すべての引数の値を合計した値です。

この構造体の，定数FUNCTIONMAX個の配列変数FInfo[]を宣言しているので，処理できる関数の数もFUNCTIONMAX（リストでは512）個になります。これ以上の関数があるとエラーメッセージを出して止まります。

1)　BAStoCする

これは簡単で，コマンドラインを作ってsystem(～)とするだけです。返り値が0以外ならエラーですので，メッセージを表示して終了します。

2)　main関数を削除 RemoveMain(　)

削除されていないファイルを読みながら削除したあとのファイルに書き込むので，まずファイル名を変更します。“foo.c”を扱う場合，“foo.c”を“foo.c‾”に変えて，“foo.c‾”を読みながら“foo.c”に書きます。

本来は各行を調べて，“void main(b_argc,b_argv) {”から，それに対応する“}”までを削除するべきです。が，中括弧の対応を調べるのが大変なので，BAStoCの癖を利用して“/******** program start ********/”から“/**************************** ****/”までを削除します。

1行ずつ処理するわけですが，行を読み込むバッファは LINELENGTH（リストでは1024）バイトです。これ以上長い行があると困ってしまうわけですが，おそらくBAStoCがそんなに長い行を作ることはないでしょう。行バッファに収まりきらないかどうかはプログラム中で検出できますが，チェックをさぼっているので長い行があると本当に困ります。

3)　コンパイル

やっていることは1)とほぼ同じです。Cコンパイラのオプション“-Fs”はアセンブラソースを作って終了させるためのものです。“-O”もつけたほうがいいかもしれません（でもXCの最適化って……）。GCCを使う場合は最低限“gcc.x -S”としてください。

4)　関数の解析 AnalyzeParameters(　)

“func”で始まる行を関数宣言だとして解析します。泥臭い字句解析をひたすら繰り返します。BASICの関数宣言は，

func［型名］関数名(引数名［；型名］［，引数名［；型名］……］)

のようになっています。あらゆるところで型名が省略でき，省略したときはintになります。

“func”を確認したあと型名があるかどうかを調べ，あった場合はそれを返り値の型とし，なかったら返り値をintとします。

関数名を変数FInfo[].fnameにコピーします。X-BASICで試してみたところ，関数名と左括弧の間にタブやスペースを入れてはいけないようですので，左括弧の直前までが関数名だと断言できます。

左括弧の次の，タブとスペース以外の文字を見て，それが右括弧以外だったら引数が存在します。

引数の処理では，まず引数名をスキップします。外部関数を作るには引数の型さえわかればいいのです。引数名の終わりは，“)”，“；”，“，”のいずれかです。その終端が“；”のときは型名がありますので，それを判別して次にあるべき“，”，“)”のところまで進みます。そして注目している文字が“)”なら終了，“，”ならば次の引数へと進みます。引数の型は順次FInfo[].param[]に貯えられます。

引数の処理が終わったら，FInfo[].param[]の終端に返り値を入れます。返り値を先頭に入れずに終端に入れるのは，この解析結果を利用する順番が引数→返り値となっているからです。

5)　ソース作成 MakeSource(　)

まずインフォメーションテーブルを適当にでっちあげます。トークンテーブルには“.dc.b '関数名', 0”という行を作ります。パラメータテーブルには「パラメータIDテーブル」のアドレスを書くだけなので，適当に識別子を作って，それを書き込みます。

ここでは“関数名_par”という識別子を作りました。パラメータIDテーブルには.dc.wでFInfo[].param[]を順番に置きます。置いていくついでに引数の個数を数え，スタック補正を計算してFInfo[].paramnとFInfo[].stackに入れます。

BASIC関数をコンパイルしたときの関数の識別子は“_関数名”になります。実行アドレステーブルには「コンパイルした関数を呼ぶ前に引数のつじつま合わせをするプログラム」のアドレスを書くので，このアドレスの識別子を“関数名”とします。

これでテーブル類は片づきました。あとはつじつま合わせのプログラムを埋め込めばいいのです。「スタックにある引数の個数だけ積み直しを繰り返す。floatなら8バイト，それ以外なら4バイトである」「bsr _関数名」「スタック補正」「返り値をメモリに置いて，そのアドレスをa0レジスタに」こういうプログラムですね。では，このとおりに組んでしまいましょう。返り値がfloatのときに注意が必要なほかは，特に問題はありません。

6)　アセンブル，リンク

“as.x -u”としています。“-u”というのは，「定義されていない識別子はリンクするときに解決するからエラーにしないでね」です。ライブラリ関数を使っているCソース（使っていないほうが珍しいが）をコンパイルしたものには必要なスイッチです。未解決な識別子の情報はオブジェクトファイルにつけられます。

リンカのスイッチ“-x”は「識別子の情報は実行ファイルに入れない」です。これ

を指定しないと，実行時にはまったく必要のない情報が実行ファイルについてしまうのです。

＊　＊　＊

ここまでの説明の方法でプログラムを組んで実行してみましょう。……はい，生成した外部関数は動きません。説明した方法は正しいものです。私も実際ここでハマってしまいました。そこで，デバッガで調べてみたところ，X-BASIC ver.2.0では，外部関数が呼ばれたとき，スタック内の引数の個数があるべきところにぜんぜん違う値が入っていることがわかりました。X-BASICのバグでしょうか。それとも「省略できる引数」があるときのみ意味を持つのでしょうか。「プログラマーズマニュアル」にそういう表記はありませんが……。

使い方

エディタでリストを打ち込み，“bas2fnc.c”という名前で保存してください。

A＞cc bas2fnc.c

とすれば“bas2fnc.x”が作成されます。“pointer type mismatch.”というwarning以外のメッセージが出たら，打ち込み間違いですので，訂正してもう1回コンパイルしてください。

使い方です。予約語の判定を小文字で行うので，“basic.cnf”内で“CAPS＝OFF”にしてください。ひとつの外部関数ファイルにしたいBASIC関数をまとめて，save@で行番号なしで保存してください。保存したファイル名を“foo.bas”とします。BASICから抜けて，コマンドシェル上で，

A＞bas2fnc foo.bas

とするとカレントディレクトリに“foo.fnc”ができます。これを“basic.x”と同じディレクトリにコピーして，“basic.cnf”に“FUNC=foo”という行を加えれば作業完了です。

●注意点

“CAPS＝OFF”にすることと，行番号なしのファイルを使うことは前述のとおりです。

内部で“BC.X”，“CC.X”，“AS.X”，“LK.X”を実行します。これらのファイルをpathの通ったディレクトリに置いてないとエラーになります。

関数内で文字表示を行っても正常動作しません。文字表示の関数はBAStoCで「OSレベルで文字表示をする関数」に変換されます。これで表示をすると，X-BASICの表示文字管理に知らされないで表示されてしまうため，カーソル位置の不一致などが起きます。

大域変数を使うには注意が必要です。bas2fncはCでmain関数になる部分を削除しますが，大域変数（BASICプログラムの先頭で宣言される変数）を残して変換します。この残った大域変数は「“foo.fnc”内にある関数でのみ有効な変数」になります。また，宣言していない大域変数を使用しても，変換途中でエラーになります。

もしメモリが足りない場合はFUNCTIONMAXの値を小さくしてください。目的のBASICファイル中にある関数の数以上であれば動作します。

課題

行番号があると変換できないのはただの手抜きです。ソースに数行加えれば対応できます。

bas2fncを使って作った外部関数をBAStoCすることはできません。DEFファイルとライブラリがないからです。プログラムができたら外部関数の分もまとめてコンパイルしてください。DEFファイルとライブラリも自動生成させるのもさほど難しくはないので，挑戦してみてください。

インフォメーションテーブルの「〜のときに呼ぶルーチン」はすべてなにもしないようになっていますね。これを，BASIC関数で特別な名前を決めておいて，それを呼ぶようにしてみるのもいいでしょう。たとえば「runしたときに呼ぶルーチン」はBASICで“func entry_run(　)”という名前にする，などです。

FUNCTIONMAXに相当するものをスイッチで指定できるようにするべきですね。またはメモリ割り当てとリスト構造で対応するなど。

＊　＊　＊

予想以上に大きなプログラムになってしまいました。引数解析の部分をもっと簡潔に書ければいいのですが……。

リスト1

```
 1: /*****************************************************************
 2:                 Oh!X 92/03
 3:                 X-BASIC の func を X-BASIC の外部関数にする
 4:                                                   by けんと
 5: *****************************************************************/
 6:
 7: #include        <stdio.h>
 8: #include        <stdlib.h>
 9: #include        <string.h>
10: #include        <io.h>
11:
12:                                         /* 定数宣言 */
13: #define FUNCTIONMAX      512            /* プログラム中のfuncの上限 */
14: #define LINELENGTH       1024           /* basicの1行の長さの上限 */
15:
16:                                         /* 関数前宣言 */
17: void RemoveMain( void );
18: void AnalyzeParameters( void );
19: void MakeSource( void );
20:
21:                                         /* global variables */
22: char BFileName[90];
23: char CFileName[90];
24: char SFileName[90];
25: char OFileName[90];
26: char FFileName[90];
27: char CommandLine[4][256] = {
28:     "bc.x ",
29:     "cc.x -Fs ",
30:     "as.x -u ",
31:     "lk.x -x -o%s %s -l doslib.l baslib.l floatfnc.l",
32: };
33: struct ptablerec {                      /* 関数の情報を格納する */
34:     char fname[64+1];                   /* 関数名 */
35:     unsigned short param[11];           /* 引数、返り値 */
36:     unsigned short paramn;              /* 引数の数 */
37:     unsigned int stack;                 /* 引数のスタック補正 */
38: };
39: struct ptablerec FInfo[FUNCTIONMAX];
40: int NumberOfFunction = 0;
41:
42:                                         /* 見ての通り main関数 */
43: void main( int argc, char *argv[] ) {
44:     if ( argc != 2 ) {
45:         printf( "Usage: bas2fnc.x <filename>.bas¥n¥n" );
46:         return;
47:     }
48:     strcpy( BFileName, argv[1] );
49:
50:     {
51:         char driven[3], pathn[90], noden[23],extn[4];
52:                                         /* フルパスなファイル名を */
53:                                         /* 各要素に分解する */
54:         strsfn( BFileName, driven, pathn, noden, extn );
55:                                         /* "foo.s" という名前を作る */
56:         strcpy( SFileName, noden );
57:         strcat( SFileName, ".s" );
58:                                         /* "foo.c" も作る */
59:         strcpy( CFileName, noden );
60:         strcat( CFileName, ".c" );
61:                                         /* "foo.o" も作る */
62:         strcpy( OFileName, noden );
63:         strcat( OFileName, ".o" );
64:                                         /* "foo.fnc" も作る */
65:         strcpy( FFileName, noden );
66:         strcat( FFileName, ".fnc" );
67:     }
68:                                         /* とにかくソースを作る */
69:     if ( system( strcat( CommandLine[0], BFileName ) ) ) {
70:         printf( "bas2fnc: bc.x を呼び出す際に"
71:                 "エラーが出ました。¥n¥n" );
72:         exit( 1 );
73:     }
74:                                         /* Cソースから main関数を */
75:                                         /* 削除する */
76:     RemoveMain();
77:
78:     if ( system( strcat( CommandLine[1], CFileName ) ) ) {
79:         printf( "bas2fnc: cc.x を呼び出す際に"
80:                 "エラーが出ました。¥n¥n" );
81:         exit( 1 );
82:     }
83:                                         /* BASICリストから引数を解析 */
84:     AnalyzeParameters();
85:                                         /* 解析結果をアセンブラ */
86:                                         /* ソースに加える */
87:     MakeSource();
88:                                         /* アセンブル */
```

```
 89:     if ( system( strcat( CommandLine[2], SFileName ) ) ) {
 90:         printf( "bas2fnc: as.x を呼び出す際に"
 91:                 "エラーが出ました。¥n¥n" );
 92:         exit( 1 );
 93:     }
 94:                                             /* リンク */
 95:     {
 96:         char clbuf[256];
 97:         sprintf( clbuf, CommandLine[3], FFileName, OFileName );
 98:         if ( system( clbuf ) ) {
 99:             printf( "bas2fnc: lk.x を呼び出す際に"
100:                     "エラーが出ました。¥n¥n" );
101:             exit( 1 );
102:         }
103:     }
104:
105:     return;
106: }
107:
108:
109: void RemoveMain( void ) {
110:     char backfile[90];
111:     char linebuf[LINELENGTH];
112:     FILE *sourcefp, *destfp;
113:
114:     strcpy( backfile, CFileName );
115:     strcat( backfile, "~" );    /* "foo.c~" というファイル名 */
116:     unlink( backfile );
117:     rename( CFileName, backfile );
118:     sourcefp = fopen( backfile, "rt" );
119:     destfp = fopen( CFileName, "wt" );
120:     do {
121:         fgets( linebuf, LINELENGTH, sourcefp );
122:         fputs( linebuf, destfp );
123:     } while ( strncmp( linebuf,
124:             "/********", 9 ) );
125:     do {
126:         fgets( linebuf, LINELENGTH, sourcefp );
127:     } while ( strncmp( linebuf,
128:             "/*******", 8 ) );
129:     while ( !feof( sourcefp ) ) {
130:         fgets( linebuf, LINELENGTH, sourcefp );
131:         fputs( linebuf, destfp );
132:     }
133:     fclose( sourcefp );
134:     fclose( destfp );
135:     return;
136: }
137:
138: int spp( char c ) {
139:     return (c==' ') || (c=='¥t');
140: }
141:
142:                                             /* 任意の個数の TAB / */
143:                                             /* SPACEを飛ばす */
144: char *SkipSpace( char *p ) {
145:     while ( spp(*p) )  p++;
146:     return p;
147: }
148:
149:
150: void AnalyzeParameters( void ) {
151:     FILE *bfp;
152:     char linebuf[LINELENGTH];   /* 1行はLINELENGTH文字未満！*/
153:     char *cp;
154:     unsigned short returnvalue;
155:     unsigned short *parameterp;
156:
157:     bfp = fopen( BFileName, "rt" );
158:     while ( !feof( bfp ) ) {
159:         fgets( linebuf, LINELENGTH, bfp );
160:         cp = linebuf;
161:         cp = SkipSpace( cp );
162:         if ( !strncmp( cp, "func", 4 ) && spp( *(cp+4) ) ) {
163:                                             /* この行は関数宣言で */
164:                                             /* あることがわかった */
165:             if ( NumberOfFunction == FUNCTIONMAX ) {
166:                 printf( "bas2fnc: 関数が多すぎます。¥n¥n" );
167:                 exit( 1 );
168:             }
169:             cp += 4;
170:             cp = SkipSpace( cp );
171:                                             /* 返り値の型名があるか */
172:             if ( !strncmp( cp, "int", 3 ) && spp( *(cp+3) ) ) {
173:                 returnvalue = 0x8001;
174:                 cp += 3;
175:             } else if ( !strncmp(cp,"str",3)&&spp(*(cp+3)) ) {
176:                 returnvalue = 0x8003;
177:                 cp += 3;
178:             } else if ( !strncmp(cp,"char",4)&&spp(*(cp+4)) ) {
179:                 returnvalue = 0x8001; /* char は int で返す */
180:                 cp += 4;
181:             } else if (!strncmp(cp,"float",5)&&spp(*(cp+5))) {
182:                 returnvalue = 0x8000;
183:                 cp += 5;
184:             } else
185:                 returnvalue = 0x8001; /* 省略時は int  */
186:             cp = SkipSpace( cp );
187:                                             /* 関数名を取得する */
188:             {                               /* '('か現れるまでコピーする */
189:                 char *funcp = FInfo[NumberOfFunction].fname;
190:                 while ( *cp != '(' )
191:                     *funcp++ = *cp++;
192:                 *funcp = EOS;
193:             }
194:             cp ++;                          /* '(' の次の文字へ */
195:             cp = SkipSpace( cp );
196:             parameterp = FInfo[NumberOfFunction].param;
197:             while ( *cp != ')' ) { /* ')'があらわれるまで */
198:                                             /* 引数名をスキップ */
199:                 while ( (*cp!=')')&&(*cp!=';')&&(*cp!=',') )
200:                     cp ++;
201:                                             /* ';'かある? */
202:                 if ( *cp == ';' ) {
203:                     cp ++;
204:                     cp = SkipSpace( cp );
205:                     if ( !strncmp( cp, "int", 3 ) ) {
206:                         *parameterp = 0x0002;
207:                         cp += 3;
208:                     } else if ( !strncmp( cp, "str", 3 ) ) {
209:                         *parameterp = 0x0008;
210:                         cp += 3;
211:                     } else if ( !strncmp( cp, "char", 4 ) ) {
212:                         *parameterp = 0x0004;
213:                         cp += 4;
214:                     } else if ( !strncmp( cp, "float", 5 ) ) {
215:                         *parameterp = 0x0001;
216:                         cp += 5;
217:                     }               /* cp は型名の次の文字 */
218:                     cp = SkipSpace( cp );
219:                     parameterp ++;
220:                 } else {
221:                     *parameterp++ = 0x0002;
222:                 }                   /* cp は ','か')' */
223:                 if ( *cp == ',' ) { /* 次の引数の頭まで飛ばす */
224:                     cp ++;
225:                     cp = SkipSpace( cp );
226:                 }
227:             }                               /* end of "while ( *cp ..." */
228:                                             /* 返り値もここにしまう */
229:             *parameterp = returnvalue;
230:             NumberOfFunction ++;
231:         }
232:     }
233:     fclose( bfp );
234:     return;
235: }
236:
237:
238: void MakeSource( void ) {
239:     char backfile[90];
240:     char linebuf[LINELENGTH];
241:     FILE *sourcefp, *destfp;
242:     int i;
243:
244:     strcpy( backfile, SFileName );
245:     strcat( backfile, "~" );    /* "foo.s~" というファイル名 */
246:     unlink( backfile );
247:     rename( SFileName, backfile );
248:     sourcefp = fopen( backfile, "rt" );
249:     destfp = fopen( SFileName, "wt" );
250:                                             /* インフォメーションテーブル*/
251:     fputs( "¥t¥t.text¥n¥t¥t.dc.l¥tnoproc¥n"
252:            "¥t¥t.dc.l¥tnoproc¥n¥t¥t.dc.l¥tnoproc¥n"
253:            "¥t¥t.dc.l¥tnoproc¥n¥t¥t.dc.l¥tnoproc¥n"
254:            "¥t¥t.dc.l¥tnoproc¥n¥t¥t.dc.l¥tnoproc¥n"
255:            "¥t¥t.dc.l¥tnoproc¥n"
256:            "¥t¥t.dc.l¥ttokentable¥n"
257:            "¥t¥t.dc.l¥tparamtable¥n"
258:            "¥t¥t.dc.l¥texectable¥n"
259:            "¥t¥t.dc.l¥t0,0,0,0,0¥n"
260:            "noproc:¥t¥trts¥ntokentable:¥n", destfp );
261:     for ( i=0; i<NumberOfFunction; i++ ) {
262:         fprintf(destfp, "¥t¥t.dc.b¥t'%s',0¥n", FInfo[i].fname);
263:     }
264:     fputs( "¥t¥t.dc.b¥t0¥n¥t¥t.even¥n", destfp );
265:                                             /* パラメータテーブル */
266:     fputs( "paramtable:¥n", destfp );
267:     for ( i=0; i<NumberOfFunction; i++ ) {
268:         fprintf(destfp, "¥t¥t.dc.l¥t%s_par¥n", FInfo[i].fname);
269:     }
270:     for ( i=0; i<NumberOfFunction; i++ ) {
271:         int j = 0;
272:         fprintf(destfp, "%s_par:¥n¥t¥t.dc.w¥t",FInfo[i].fname);
273:         FInfo[i].stack = 0;
274:         FInfo[i].paramn = 0;
275:         while ( FInfo[i].param[j] < 0x8000 ) {
276:             fprintf( destfp, "%u,", FInfo[i].param[j] );
277:             FInfo[i].paramn ++;
278:             FInfo[i].stack +=
279:                     (FInfo[i].param[j++]==0x0001) ? 8 : 4;
280:         }
281:         fprintf( destfp, "%u¥n", FInfo[i].param[j] );
282:     }
283:                                             /* 実行アドレステーブル */
284:     fputs( "exectable:¥n", destfp );
285:     for ( i=0; i<NumberOfFunction; i++ ) {
286:         fprintf( destfp, "¥t¥t.dc.l¥t%s¥n", FInfo[i].fname );
287:     }
288:                                             /* 各関数エントリ */
289:     for ( i=0; i<NumberOfFunction; i++ ) {
290:         char *fn = FInfo[i].fname;
291:         fprintf( destfp, "%s:¥n¥t¥tmovea.l¥tsp,a6¥n"
292:                 "¥t¥tmove.w¥t#%u,d1¥n"
293:                 "¥t¥ttst.w¥td1¥n"
294:                 "¥t¥tbeq¥t?%s0¥n"
295:                 "¥t¥tmove.w¥td1,d0¥n"
296:                 "¥t¥tsubq.w¥t#1,d0¥n"
297:                 "¥t¥tmulu.w¥t#10,d0¥n"
298:                 "¥t¥taddq.l¥t#6,d0¥n"
299:                 "?%s2:¥n¥t¥tmove.l¥t6(a6,d0.l),-(sp)¥n"
300:                 "¥t¥ttst.w¥t(a6,d0.l)¥n"
301:                 "¥t¥tbne¥t?%s1¥n"
302:                 "¥t¥tmove.l¥t2(a6,d0.l),-(sp)¥n"
303:                 "?%s1:¥n¥t¥tsub.l¥t#10,d0¥n"
304:                 "¥t¥tsubq.w¥t#1,d1¥n"
305:                 "¥t¥tbne¥t?%s2¥n"
306:                 "?%s0:¥n¥t¥tbsr¥t_%s¥n"
307:                 "¥t¥tlea¥t%u(sp),sp¥n"
308:                 "¥t¥tlea¥t?%sret(pc),a0¥n",
309:                 fn, FInfo[i].paramn, fn, fn, fn, fn,
310:                 fn, fn, fn, FInfo[i].stack, fn );
311:         if ( FInfo[i].param[FInfo[i].paramn] == 0x8000 ) {
312:             fputs( "¥t¥tmove.l¥td1,6(a0)¥n"
313:                    "¥t¥tmove.l¥td0,2(a0)¥n", destfp );
314:         } else {
315:             fputs( "¥t¥tmove.l¥td0,6(a0)¥n", destfp );
316:         }
317:         fprintf( destfp,"¥t¥tmoveq.l¥t#0,d0¥n"
318:                 "¥t¥trts¥n"
319:                 "?%sret:¥n¥t¥t.dc.w¥t0¥n¥t¥t.dc.l¥t0,0¥n", fn );
320:     }
321:
322:     while ( !feof( sourcefp ) ) {
323:         fgets( linebuf, LINELENGTH, sourcefp );
324:         fputs( linebuf, destfp );
325:     }
326:     fclose( sourcefp );
327:     fclose( destfp );
328:     return;
329: }
```

BASIC

圧縮したデータをBASICで使う
LHAFNC.FNC

Kamiyama Mitsuru 紙山　満

ゲームなどで使用できる便利な外部関数です。LHA.Xを使用して作成した圧縮データファイルをメモリ上の配列に展開することができます。ライブラリも掲載しますので活用してください。

僕は昔，BASICで大きなRPGを作っていました。しかしマップのファイルサイズがとんでもなく大きくなって困ってしまいました。

たとえばですが，ドラクエのようなマップを考えてみましょう。フィールドを256×256マスの大きさにするとしてBGを使うとすると，1マスが2バイト（スプライトナンバーとパレットナンバー，反転の有無などの情報）になりますから，256×256×2でなんと131072バイト！　これに裏のフィールドや町，ダンジョンなど入れようものなら，それだけでディスクが一杯になってしまいます。

そこで頭のいい(？)僕はあるアイデアを思いつきました。それはLH.XやLHA.Xで圧縮したファイルをBASIC上で解凍できるようにして，しかも解凍したデータをディスクに書き込むのではなく，BASICの配列に流し込むことはできないだろうか？ということです。

というわけで，さっそくLHA.Xの自己解凍プログラムの部分を解析して改造を施し，BASICの外部関数にしてみました。これでたくさんのデータを持つようなプログラムでも少量のディスクで動作させることができます。

そして，コンパイル用のライブラリも用意しました。

```
cc ????.bas lhafnclib.o
```

これだけで，lha()を使ったプログラムもコンパイル可能！　もちろん，Cやアセンブラからも呼び出せます。

いつもはライブラリ兼用になるようにプログラムを組むのですが，ややこしくなるので，今回は別のファイルにしました。といっても，ほとんど同じなのでlhafnc.sをちょこちょこいじるだけで大丈夫です。

なおライブラリのほうは，エラーチェックをしていませんので，BASIC上でちゃんと動くようになってからコンパイルしてください（ライブラリからは配列の型がわからない）。

入力&操作方法

ソースリストを入力してアセンブル，リンクしてもらってもいいのですが，普通の人にはダンプリストのほうが確実に入力できるでしょう。プログラムはリスト1です。このリストはLHAで圧縮されていますので1992年6月号の付録ディスクに収録されていたMAC.Xなどのマシン語入力ツールを使って打ち込んでください。これを2063バイトでセーブします。

リスト2はC用のライブラリです。これも同様に入力し，2024バイトでセーブしてください。これらはLHAによって圧縮されたものですから，

```
LHA E LHAFNC.LZH
```

などのようにして展開してください。できあがったLHAFNC.FNCはBASICと同じディレクトリに，LHAFNC.Oは環境変数libの示すディレクトリに入れておいてください。

それでは使い方です。

まずは，例によってBASIC.CNFに，

```
FUNC=LHAFNC
```

の1行を加え，X-BASICを起動。……と，その前に適当なファイルをLHAで圧縮しておきましょう。たとえば，

```
TEST.DAT　1024 バイト
```

があって，これを圧縮して，

```
TEST.LZH　128 バイト
```

となったとしましょう。

さて，これをX-BASICで解凍するには，

```
10 char buf(1023)
20 lha("test.lzh",buf)
```

とするだけ。これで，bufにはTEST.DATの内容が入っているはずです（当然ですが，ディスクにTEST.DATが作成されるようなことはありません）。

ここで注意してほしいのが，配列の大きさについてです。これは解凍されるデータの分以上とっておかなければなりません。TEST.LZHの大きさではなく，圧縮する前のTEST.DATの大きさです（まあ，当たり前かもしれませんけど）。もちろん，配列の型はcharの必要はありません。

```
10 int buf(255)
10 float buf(127)
```

でも構いません（str型は使えません）。

配列の大きさが足りない場合や，ファイルがオープンできないときはエラーが出ます。LHAなどで圧縮されたもの以外のファイルは指定しないでください。また，たくさんのファイルをまとめて圧縮したものを指定しないでください。

```
×lha a test.lzh abc.dat def.dat
○lha a test.lzh abc.dat
```

正常終了のときは，戻り値には解凍されたときのファイルのサイズ（上の例の場合は1024）が入ります。

LHAに感謝

というわけで，いかがでしょうか。用途としては，やはりデータ量のはっきりしたゲーム関係が中心でしょうか。ぜひ，ゲームを作る際に活用してください。

あ，そうそう。*.lzhをリネームして*BGDとかにしておけば，間違って解凍されることもないし，このプログラムを使っていることもバレなくていいんじゃないかと思います。

```
20 lha("field1.BGD",buf)
```

ってな感じで。

なお，ディスアセンブルにはDIS.Xを使い，解凍ルーチンはLHA.Xをもとに改造しました。このプログラムができたのもLHAという優秀なツールがあったからです。原作者の吉崎氏やX68000への移植を行った岡田氏に感謝します。

リスト1 LHAFNC.LZH

```
000000  23 C0 2D 6C 68 35 2D E9  : 2F
000008  07 00 00 12 0C 00 00 00  : 25
000010  60 27 1A 20 01 0A 4C 48  : 60
000018  41 46 4E 43 2E 46 4E 43  : 1D
000020  66 24 48 00 00 06 DE 73  : 29
000028  A3 DA 34 DB 92 7D CC 63  : CA
000030  3B 07 42 72 21 1E 68 0D  : AA
000038  0E 4C 10 9A 68 06 F0 49  : AB
000040  31 26 92 46 DC 91 4E 8C  : 76
000048  33 74 6C B1 33 63 54 C0  : 6E
000050  67 32 EA 22 34 20 14 AD  : BA
000058  0B 66 73 B4 A2 A0 AA 2D  : B1
000060  D3 C1 14 53 65 55 4B 65  : 65
000068  24 AB 54 59 AA 45 22 16  : A3
000070  4C 36 95 4B 2C BB 39 35  : B7
000078  D7 15 4C AC 0A 4D 43 49  : C7
------------------------------------
  SUM:  0D 67 07 38 E8 82 12 BF  1433

000080  29 26 48 55 17 67 4E FB  : B3
000088  EF F8 2D C1 B6 4B 6F 0E  : 53
000090  F1 6F 1D DE EA E3 6D B9  : 4E
000098  32 7E 38 7F 8F D3 B8 42  : C3
0000A0  0E 28 1D 5C 78 01 C7 87  : 76
0000A8  D5 F9 D7 FF F5 E0 FA 98  : 0B
0000B0  FB 03 FE AE 3E BE BA B9  : 19
0000B8  B5 5F 15 80 1E 1F 6A 41  : 91
0000C0  C3 87 92 F3 24 60 BC 53  : 62
0000C8  07 1A 9B C2 08 08 FD E4  : 6F
0000D0  B9 40 CE 9B AC 38 65 B2  : 5D
0000D8  C9 28 A8 B0 C6 07 2D 80  : C3
0000E0  F8 12 2B 1B 03 24 1D 81  : 15
0000E8  D1 67 44 2C 62 B0 D0 C3  : 4D
0000F0  FE 78 74 D2 8C BB FB 84  : 82
0000F8  3D 88 74 35 7E 59 0D DF  : 31
------------------------------------
  SUM:  1E 10 CB 4A 1C B5 07 2D  866F

000100  CC 53 1F 10 83 CC 7D 7C  : 96
000108  9E 88 78 9F 72 46 20 8A  : 9F
000110  E3 7C 4E 15 B8 36 8F EB  : 2A
000118  49 57 AF 74 C6 18 1D 65  : 23
000120  C8 F9 80 E9 DC DF C2 09  : B0
000128  4E 60 4E 7C 49 C4 63 3A  : 22
000130  EB D3 3A C7 F6 AD F2 D7  : 2B
000138  4A 81 38 F6 71 4D 2E 38  : 1D
000140  9F 43 E0 E1 92 DF 5D 6A  : DB
000148  3B 27 EC E5 54 84 50 39  : 94
000150  28 18 4E 19 38 9E D7 2F  : 83
000158  C3 7F E1 09 D7 64 93 9C  : 96
000160  8B 60 9A CF C3 10 B0 30  : 07
000168  4E 10 A2 94 AF FC 54 EC  : 7F
000170  F4 D1 45 CE CE 1A 38 38  : 30
000178  58 C3 EA 9F D9 E8 29 56  : E4
------------------------------------
  SUM:  CB 60 3A 12 0D 70 0A C0  7696

000180  67 EF 6D 0A 30 AA 51 AA  : A2
000188  54 78 D9 57 57 2E 6A B3  : 9E
000190  B0 C6 95 6E D4 61 18 37  : FD
000198  12 80 E7 97 31 50 B3 91  : D5
0001A0  14 F8 32 22 8C 9E 1E 65  : 0D
0001A8  79 28 F8 70 02 92 A9 65  : AB
0001B0  BB 7E D5 15 50 B5 FE E5  : 0B
0001B8  16 E6 B7 AA 07 6D AF 5E  : DE
0001C0  0E 57 9B 97 33 76 A0 C9  : A9
0001C8  BD 14 C0 4A 8B 28 01 74  : 03
0001D0  FA 3C 00 05 96 7B BE 13  : 1D
0001D8  FE 4A 82 C7 5A 76 30 0D  : 9E
0001E0  25 3B 6D 7F 06 45 31 0E  : D6
0001E8  32 03 34 93 B5 46 36 BC  : E9
0001F0  98 6C 46 E6 F3 1C EF 88  : B6
0001F8  FE 3F 96 4D 91 77 8E 54  : 0A
------------------------------------
  SUM:  8B 0B D2 A9 5E 88 6D 35  B227

000200  3C 10 C3 5A 5C 63 A1 8C  : 55
000208  37 BF 35 19 14 76 6F BC  : F9
000210  8B 0E E6 DC 1F 0D 8F 0E  : 24
000218  84 6B 31 8E EC DF 5F 3D  : 15
000220  E3 79 53 E9 E8 C8 0E 1C  : 72
000228  DA DC 2B 27 F6 1D C0 39  : 14
000230  F5 F5 4F 00 02 40 65 64  : 44
000238  8B 1F BB 6E CA A4 99 FD  : D7
000240  D7 0C 93 2B A7 9B B3 59  : EF
000248  FF 21 90 72 A4 45 61 0F  : 7B
000250  99 6A 0C 52 7A 6C F7 72  : B0
000258  8A F5 D2 34 5E 37 77 DD  : 6E
000260  2A 36 EF E3 30 74 4D 24  : 47
000268  F2 F8 E5 A1 1C D6 D5 ED  : 24
000270  D7 6C 8B B4 11 15 FE A4  : 4A
000278  27 EF 7C DB 78 83 14 FE  : 7A
------------------------------------
  SUM:  D2 C6 73 91 1D F3 80 B3  4112

000280  74 1D 02 D6 D4 8C E5 64  : 12
000288  4C 00 F3 F3 54 9F 71 3D  : D3
000290  6A 13 5D 08 4F D7 59 83  : E4
000298  E6 C3 F7 EA 01 8C 91 57  : FF
0002A0  F9 F9 D2 6D 40 E9 F6 D3  : 23
0002A8  DC DC 5E 6C 88 3E 64 19  : C5
0002B0  2C 38 BF 89 66 7A 8E 83  : 9D
0002B8  69 66 EF 61 66 07 16 C7  : 69
0002C0  DC 5B E1 87 CD 31 58 50  : 45
0002C8  86 6F 69 5C FA F8 B5 2A  : 8B
0002D0  9E 8B A6 8D 6A C0 F2 BC  : 34
0002D8  FB 98 C5 A8 BD B4 7C 7C  : 69
0002E0  1B 8A 0D 04 F4 27 A2 32  : A5
0002E8  73 5A 01 0F C4 E7 76 21  : 1F
0002F0  EF 0A 9B 07 DE 43 EB E7  : 8E
0002F8  13 50 4A B8 F2 E2 58 6C  : FD
------------------------------------
  SUM:  05 91 CF 68 82 06 14 09  CE10

000300  48 33 52 E1 3F 82 E9 79  : D1
000308  8C C4 E6 8C 19 B3 07 70  : 05
000310  31 3F 78 69 AA 03 2F FB  : 28
000318  2E 38 0B 0F E3 17 62 60  : 3C
000320  CF 89 DE C1 1D 2D 5D FD  : 9B
000328  85 07 6D 3B E2 A7 75 01  : 33
000330  FD 2B F4 93 D0 4D CB 90  : 27
000338  02 47 3B 5A D4 16 81 09  : 52
000340  F9 22 F0 D4 BC 0A 02 2E  : D5
000348  0E 0E 17 DE 04 27 E9 52  : 77
000350  D4 52 94 2F F4 AE E7 27  : 99
000358  2E 41 25 F0 54 72 E4 F0  : 1E
000360  D8 74 AC 8A EC 17 71 36  : 2C
000368  98 3B F2 60 8D 79 3A DE  : 43
000370  82 0C 8A 81 2B 6A 0C 35  : 6F
000378  A1 3F BD 3B 47 A5 2D 52  : 43
------------------------------------
  SUM:  22 2D DA 45 7B 76 39 0D  DD21

000380  6C A7 A3 4B 49 C7 33 E4  : 28
000388  CC 9F AC 27 FE 18 4F D0  : 73
000390  74 76 B5 B0 7C FC C7 AE  : 3C
000398  94 DF B5 34 46 F7 07 94  : 34
0003A0  7C 1C 16 72 AC 7D B5 A6  : A4
0003A8  7D 09 3E C4 27 E4 75 10  : 18
0003B0  5E 61 5D 0D C9 90 C7 65  : AE
0003B8  BD 76 F5 FD B9 59 1E ED  : 42
0003C0  D0 06 1E CC 7B A0 9E C4  : 3D
0003C8  7A BC FE 55 17 58 67 FA  : 59
0003D0  80 BC 05 E4 13 47 E3 1E  : 80
0003D8  B0 19 FF C7 93 6C F1 4A  : C9
0003E0  1F A1 E2 F2 13 FB FA 1F  : BB
0003E8  7E 40 B7 69 20 49 D7 2E  : 4C
0003F0  3B 2F 60 DE 13 FE 48 CB  : CC
0003F8  E2 5D 8D F9 D1 81 EC 91  : 94
------------------------------------
  SUM:  88 9B 05 94 AD 8A 3D CD  A3A3

000400  82 64 8C 5B EE 14 12 B6  : 97
000408  03 3F D6 21 BF 8C 03 16  : 9D
000410  8B 7F D2 C4 D9 93 47 60  : B3
000418  63 1D 99 92 81 63 5E 84  : 71
000420  9D AC F8 79 23 3D 31 B7  : 02
000428  E8 D8 77 2B B5 76 34 A2  : 63
000430  14 8E 2A CA F7 9D EC C2  : D8
000438  7F 8F EB 01 F6 C1 32 EF  : D2
000440  8E 68 D4 69 5E F4 B4 DB  : 14
000448  B5 4A 57 4B C5 5E CD F0  : 81
000450  54 35 4B 0F F5 49 F9 ED  : 07
000458  AB FE 09 43 4A F4 C1 1C  : 10
000460  8E C4 41 E7 95 7D 33 8A  : 49
000468  BD 90 D8 90 09 C5 29 2B  : D7
000470  0D 4F 74 39 11 18 C0 75  : 67
000478  0F 55 C0 7D F9 05 72 4E  : 5F
------------------------------------
  SUM:  34 BD 1D 74 D6 95 06 06  5C64

000480  B4 2B BA 3B 51 F7 54 A2  : 12
000488  32 9B 01 26 94 27 F7 80  : 26
000490  F6 40 B8 64 33 CB B1 EB  : EC
000498  0B 87 A4 BC EB DA 18 B7  : 86
0004A0  D2 C4 8C 1F EE 7B E0 AF  : 39
0004A8  3D 9E 58 DD C4 18 C4 FD  : AD
0004B0  A9 E4 0B 62 E5 3D B8 65  : 39
0004B8  52 26 F9 F9 45 B7 3B 56  : F7
0004C0  55 34 A4 5D EE 14 5B 94  : 7B
0004C8  F4 58 03 52 E2 3F 17 B5  : 8E
0004D0  85 4D 96 43 E9 9B A7 5B  : 31
0004D8  B0 FE F6 4D 4B 85 F9 6E  : 28
0004E0  D7 DF 0E 15 06 68 F3 F7  : 31
0004E8  32 9B DB 22 63 C4 AB A5  : 41
0004F0  CB F1 0D D1 1D E0 56 1B  : 08
0004F8  8F 18 C0 8B 4F D2 03 B1  : C7
------------------------------------
  SUM:  D2 53 E8 AA B8 9B B4 A5  ED06

000500  DF B2 01 9D C8 F5 DD 8F  : 58
000508  35 7E 0A DC B2 32 FA F9  : 70
000510  22 DC DC 6F 5E 61 27 DC  : 0B
000518  37 48 DC 68 FA 25 C4 FE  : A4
000520  74 46 52 CC 32 68 F9 F9  : 64
000528  A3 AF B3 F6 78 D3 26 98  : 04
000530  F5 EE 2B 7A F2 4C D4 64  : FE
000538  71 86 86 DF 86 F4 D5 5F  : 0A
000540  3B CA 29 D4 77 86 80 D4  : 53
000548  F9 AA 8A 5D 89 C5 3D 1D  : 32
000550  B8 0A D9 21 FC 7C B0 AE  : 92
000558  32 5C 70 A2 21 E8 95 0B  : 49
000560  CA 9B E5 51 73 E6 FB F7  : E6
000568  5D FA EC D0 8E 2B EB 43  : FA
000570  12 27 91 16 83 C2 CF 39  : 2D
000578  96 E3 AA 49 3F 25 CA FC  : 96
------------------------------------
  SUM:  D7 36 81 DF D4 CF 0B CF  5486

000580  96 E7 27 F2 5D 66 85 6F  : 4D
000588  EC 26 28 CF BE 22 72 9B  : F6
000590  A8 1F 0A 34 63 79 52 B7  : EA
000598  66 21 71 15 90 86 D1 4F  : 43
0005A0  07 D7 C0 0B 45 96 A9 9E  : CB
0005A8  87 F1 3C A6 46 99 88 93  : 54
0005B0  37 51 FE 4E BA 8E E0 C8  : C4
0005B8  AC 59 E5 05 A4 0C 4C 8F  : 7A
0005C0  6A E3 4C A3 15 B5 0B 79  : 8A
0005C8  AD AE C1 1B D3 8B 4D EF  : D1
0005D0  31 EC 93 8E 8B C8 50 15  : F6
0005D8  EA 2B 13 56 F4 3D 81 9C  : CC
0005E0  E2 24 BE D9 5E 63 B7 BB  : D0
0005E8  04 28 37 8D 81 23 A5 C8  : 01
0005F0  81 18 F0 80 A9 1F B8 D2  : 5B
0005F8  31 DF C1 64 9B CF 1A F4  : AD
------------------------------------
  SUM:  CB AA 02 FA 81 09 CE FA  65E2

000600  EB C0 E3 16 5B AB 27 BC  : 8D
000608  45 CA F2 36 1A F5 8D 06  : D9
000610  A1 AA DC EC FF CC 94 DC  : 4E
000618  BB 3D DC A8 0A C7 44 5C  : ED
000620  BA 51 D1 E7 62 D5 71 96  : 01
000628  D5 BD 3B 27 32 E2 AE FC  : B2
000630  0C AF E8 6B AB 8B 57 7A  : 15
000638  8A 53 4D EA D6 D5 EF 7A  : 28
000640  6E 65 24 86 32 B7 87 67  : 54
000648  F4 6B 32 E0 2B 93 99 13  : DB
000650  BA 4C 1B 8D D2 4C 25 C8  : B9
000658  49 D7 66 61 2B C7 44 9C  : B9
000660  A4 86 E7 2C D2 2F 88 CC  : 92
000668  7B A6 A8 76 B0 47 85 37  : F2
000670  2D 02 9C 0C CD F5 22 A5  : 60
000678  78 73 74 3B FD 1E 9A B1  : 00
------------------------------------
  SUM:  DA 15 44 80 39 30 43 B7  BA70

000680  59 DE 1D DF E8 EB B1 74  : 2B
000688  C2 84 A7 FE 9A AA 6A D5  : 6E
000690  8D B1 8D B7 C6 C5 8E 70  : 0B
000698  C3 F3 05 56 66 7F E8 2C  : 0A
0006A0  38 01 FA B0 AD C0 3E 5E  : EC
0006A8  58 DE B2 A2 68 EF 2E 42  : 51
0006B0  A9 D2 DA 91 40 33 65 C8  : 86
0006B8  AF 34 6F 5D BC 3D 7A 6B  : 8D
0006C0  E8 0E A9 C7 97 5A 8E 2F  : 14
0006C8  F1 CB C9 D3 5B 52 E1 2A  : 10
0006D0  76 39 19 82 BB F9 6B 19  : 82
0006D8  24 C7 61 E2 51 5E 46 65  : 88
0006E0  B6 A8 EF 6F 09 2A B5 B5  : 59
0006E8  36 58 B5 0C 75 E7 FE 80  : 29
0006F0  A6 DD 85 0F 66 55 38 A0  : AA
0006F8  F3 31 60 AB 1D 6E A2 B1  : 0D
------------------------------------
  SUM:  4B D2 C0 5D BE CF 89 15  2634

000700  60 5E 0F C4 AE 5C 8A 77  : 9C
000708  5F C2 61 75 66 58 DA BA  : 49
000710  FD F3 0B AB 97 BE 32 87  : B4
000718  BC B3 79 F7 5C F9 D0 3C  : 40
000720  E4 A6 A5 7A CB 4C 8B 3F  : 8A
000728  58 89 5B D7 66 EB 7E 1A  : FC
000730  1F 4E 5D B4 7E CC AD 0C  : 81
000738  4B 1E BA DD 84 F3 1C ED  : 80
000740  40 7F 6F 10 6F 4E 16 51  : 62
000748  5E 91 E8 CE C3 4E BB 55  : C6
000750  6F 78 B5 13 FE 7E 4A 72  : E7
000758  6F 64 BA 93 C4 93 88 AF  : AE
000760  4A 4F 23 07 EF 49 BF 93  : 4D
000768  F8 D5 EA AB EA AB D6 57  : 24
000770  D4 57 A3 82 EE 4F 2A 4D  : 04
000778  B1 20 AB D2 F4 7D 55 74  : 88
------------------------------------
  SUM:  61 E8 2C 47 E9 CE EF B8  3EAB

000780  95 FE 4A E4 AB FD 70 5E  : 37
000788  19 61 8F FC FE 90 FC 9A  : 29
000790  59 0F F3 C4 73 A0 D4 D8  : DE
000798  4B 3D 00 E5 40 4E B2 06  : B3
0007A0  82 3B 20 DD 05 08 33 0A  : 04
0007A8  21 C7 3E 28 27 8F 4A CC  : 1A
0007B0  8C 14 41 B1 B3 B2 13 28  : 32
0007B8  98 AB 2D 76 7B 51 CA DC  : 58
0007C0  C7 D9 05 40 AD F0 84 9E  : A4
0007C8  E8 5A 28 0E 3A E1 F9 9A  : 26
0007D0  B5 1C 34 43 D0 A5 08 8D  : 52
0007D8  01 49 A7 14 1E 11 9C 33  : 03
0007E0  A1 48 28 64 D3 28 6E 7C  : 5A
0007E8  C1 3C 4C 9E 6C AB 97 44  : D9
0007F0  50 69 CC F9 6E 3E 94 2B  : E9
0007F8  52 25 DB 0A CE EB 5E 21  : 94
------------------------------------
  SUM:  82 16 BB 5F 06 98 64 B4  C6EE

000800  72 6F 5A 0E 8E 27 43 80  : C1
000808  15 86 93 67 C1 A0 00 00  : F6
000810  00 00 00 00 00 00 00 00  : 00
000818  00 00 00 00 00 00 00 00  : 00
000820  00 00 00 00 00 00 00 00  : 00
000828  00 00 00 00 00 00 00 00  : 00
000830  00 00 00 00 00 00 00 00  : 00
000838  00 00 00 00 00 00 00 00  : 00
000840  00 00 00 00 00 00 00 00  : 00
000848  00 00 00 00 00 00 00 00  : 00
000850  00 00 00 00 00 00 00 00  : 00
000858  00 00 00 00 00 00 00 00  : 00
000860  00 00 00 00 00 00 00 00  : 00
000868  00 00 00 00 00 00 00 00  : 00
000870  00 00 00 00 00 00 00 00  : 00
000878  00 00 00 00 00 00 00 00  : 00
------------------------------------
  SUM:  87 F5 ED 75 4F C7 43 80  5277
```

リスト2 LHALIB.LZH

```
000000   21 04 2D 6C 68 35 2D C4   : 4C
000008   07 00 00 2A 0C 00 00 72   : AF
000010   25 27 1A 20 01 08 6C 68   : 63
000018   61 6C 69 62 2E 6F 5B F3   : 83
000020   48 00 00 06 B8 73 C3 D6   : 12
000028   34 DB 8B FF E7 CD B6 C0   : C3
000030   68 3C 04 1C 23 09 4E CD   : 0B
000038   C8 61 18 06 F0 12 41 DD   : 67
000040   2D 8C 76 C5 3A 30 CD C1   : EC
000048   B3 54 CD 8D 33 60 C6 E6   : A0
000050   50 45 69 C6 0A 02 16 CE   : B4
000058   32 D2 C4 88 B2 35 5B E0   : 72
000060   88 46 B6 2A 96 8A C8 B6   : 4C
000068   88 B3 64 8C 54 2C 98 69   : AC
000070   62 B6 D9 76 72 6B AE 99   : 8B
000078   54 B4 0A 65 89 A4 B4 B3   : 0B
--------------------------------------
  SUM:   82 69 C4 70 63 93 C2 91   B531

000080   E0 15 45 C9 E3 BF F7 77   : 13
000088   73 44 AB 5D BE 3D BC 3A   : B0
000090   C6 DB 6D E9 A0 00 CA BF   : 20
000098   07 95 0E 2E 15 FE 0F E1   : DB
0000A0   BE 09 84 00 A0 D8 93 81   : D7
0000A8   CD 92 01 31 60 8C 5B 12   : EA
0000B0   1C 98 63 CC E0 25 B3 1B   : B6
0000B8   EE 67 30 26 2D 10 A1 CC   : 55
0000C0   93 0E FF 96 1D 72 1B 9F   : 7F
0000C8   E0 13 E0 03 D2 E6 52 3B   : 1B
0000D0   D0 76 B1 2E 27 36 F9 81   : FC
0000D8   6B B3 87 95 0A 5C D8 A5   : 1D
0000E0   CD 79 57 21 AF 59 E4 36   : E0
0000E8   FE 29 75 58 53 A1 53 59   : 94
0000F0   62 A9 CD 75 CF 7B 9B B4   : E6
0000F8   7B 95 0F C8 F7 0C F5 11   : F0
--------------------------------------
  SUM:   0B 8D 42 72 4B FE D3 1F   CE23

000100   26 EE A2 7E 89 0D 70 F1   : 2B
000108   3D F3 80 1D C6 6E E5 44   : 2A
000110   EA E4 7C AB D7 D6 93 12   : 47
000118   EA 05 61 82 B6 10 12 EF   : 99
000120   8F F4 0F 72 05 19 6A BF   : 4B
000128   10 1E C3 1D 87 C6 84 45   : 24
000130   B4 2E BA 44 BA FE 08 FC   : 9C
000138   F6 D3 1F B0 78 C2 2F 7F   : 80
000140   8C 91 1A CA 42 93 88 E5   : 43
000148   43 DA 74 6A AE DF 33 99   : 54
000150   60 11 80 95 6F F2 5B AF   : F1
000158   6F 45 3C 5B 04 78 8F BE   : 14
000160   60 A1 B7 4C D5 99 5F 16   : E7
000168   B8 0E D5 E7 8F 95 89 79   : A8
000170   20 F9 29 F3 3F F8 52 6A   : 28
000178   A0 52 B8 F1 F1 AA 81 93   : 4A
--------------------------------------
  SUM:   F6 98 61 86 91 AC 7F 2C   9971

000180   96 3F F1 7F D3 C0 8B 64   : C7
000188   A3 1B 4A 5D E3 2A A0 B1   : C3
000190   8A D4 FA AC BD 1C 1E 53   : 4E
000198   8B 1D 4E 7B 20 0B 58 AB   : 9F
0001A0   23 85 39 9E 32 70 BD 42   : 20
0001A8   D2 46 65 79 2A 33 18 12   : 7D
0001B0   A3 0B D6 1E 85 C2 67 3B   : 8B
0001B8   F2 D2 E3 85 5D 8A 28 FC   : 37
0001C0   5D 96 96 22 A3 FB 6D 2A   : E0
0001C8   2D 3D A4 84 F5 4B 50 EC   : 0E
0001D0   40 BB A8 70 BA 18 2F 68   : 7C
0001D8   D2 70 1E D3 44 02 79 BB   : AD
0001E0   0A AC 70 04 51 78 E5 24   : FC
0001E8   FF AA 5A 08 A2 97 46 86   : 10
0001F0   38 25 15 48 B3 F4 94 E8   : DD
0001F8   12 9B 27 CF A6 53 99 39   : 6E
--------------------------------------
  SUM:   C7 07 E0 C9 B3 B6 C2 A2   32AD

000200   18 FE 9B 13 45 15 6C 3E   : C8
000208   44 25 A3 40 1F 91 C3 95   : 54
000210   0A C9 9D 8A A2 D4 E1 63   : B4
000218   D3 0A D4 AB A1 06 DC DF   : BE
000220   25 58 97 44 26 2C DE F2   : 7A
000228   6E 7B D4 80 F7 E8 76 69   : FB
000230   B6 D4 AD A3 DE 0B D3 78   : 0E
000238   F7 0D E7 17 97 46 86 BC   : 21
000240   A0 E2 2E 47 CB BB 7D 73   : 6D
000248   DE 01 A0 BB 0E 18 04 A0   : 04
000250   D1 D0 49 1A D0 A7 5F AF   : 89
000258   B0 72 AF 36 EB D0 4E 90   : A0
000260   E8 F7 26 4A 92 B7 F3 AA   : 35
000268   63 45 A3 2A 32 C8 59 90   : 58
000270   C1 69 EF 27 83 C2 98 F7   : 14
000278   0D 0D 7A BA 95 5E 9E 13   : F2
--------------------------------------
  SUM:   91 81 A6 AD A9 CE 49 3A   4FF0

000280   42 91 48 CB 57 3F 8E F4   : FE
000288   A4 52 03 F5 8A 44 A5 29   : 8A
000290   95 27 57 5C 57 AA C0 EF   : 1F
000298   5B F0 CC EA 0A 6F 3A 0B   : BF
0002A0   E7 9E 8D FC 41 77 7A 6C   : AC
0002A8   9D 02 37 D5 6C E6 67 CE   : 32
0002B0   31 8C AA 31 85 74 6D C5   : C3
0002B8   F7 3B D7 82 FA 7A 17 7F   : 95
0002C0   03 D1 85 F0 D6 82 DC 81   : FE
0002C8   EB FE F3 32 21 81 28 EB   : C3
0002D0   A8 BA 3D 0E 8C A8 1E 84   : 83
0002D8   0C 85 9E 6F E3 45 9E CB   : 2F
0002E0   A0 FF A8 1C 2D C4 0B 67   : C6
0002E8   FE 6D D7 BC 8F F6 85 D1   : D9
0002F0   48 4C 18 23 0B EA 6D 9E   : CF
0002F8   E1 9A C8 D0 D7 75 52 D6   : 87
--------------------------------------
  SUM:   EB C1 65 F4 72 F0 A1 FC   1A37

000300   C8 18 A7 4B 34 3D 32 91   : 06
000308   3B 15 59 9C 5F AD 9D CE   : BC
000310   C0 A6 D6 43 5A 8A 22 F7   : 7C
000318   16 80 10 BC 4E 2D 47 CF   : F3
000320   80 4E D6 3D E4 BD C3 6E   : B3
000328   74 0C 55 1F 1A 58 97 2A   : 27
000330   BB A7 76 60 BE 80 EA 7C   : DC
000338   0B 72 90 73 92 F2 B9 2F   : EC
000340   34 E2 DF F8 06 3A B2 17   : F6
000348   FF F6 B7 E1 A2 3F 92 0E   : 0E
000350   B5 2F 34 62 37 B4 61 A9   : 6F
000358   DA 0D 36 69 4E 8E 09 D1   : 3C
000360   A8 C7 7D E5 68 7F 9B 8E   : E1
000368   FA 1A C8 38 70 00 62 F2   : D8
000370   6D C3 04 5F C1 7C 99 6D   : D6
000378   AE 69 6A A4 05 31 85 83   : 63
--------------------------------------
  SUM:   12 E7 CA D9 54 0F FE 77   8EA4

000380   72 0B 17 22 0B E3 EB 2B   : BA
000388   6B 0C D5 EE 21 58 56 86   : 8F
000390   A2 13 65 26 67 67 9A 5A   : 02
000398   57 0E 18 82 AB D4 4A AA   : 72
0003A0   37 83 98 5B 4F A5 1A E0   : 9B
0003A8   C8 D1 96 24 BF A6 81 2A   : 63
0003B0   B4 42 C3 05 9B 30 5F 49   : 31
0003B8   26 BF 56 6C B3 29 47 16   : E0
0003C0   B5 7C 65 58 9F 29 24 8D   : 67
0003C8   00 91 FC 10 5F 32 74 4D   : EF
0003D0   BB E1 97 8F 1D 45 F2 5A   : 7[illegible]
0003D8   A4 DB BC E7 79 EC 60 73   : 5A
0003E0   99 CC BA F7 11 9D 3C A7   : A7
0003E8   AB 05 F1 86 94 27 BF BB   : 5C
0003F0   D4 C9 DC 24 4E 37 8F EE   : 9F
0003F8   D7 DC 7C 53 33 FD FC 10   : BE
--------------------------------------
  SUM:   B2 CC 67 7A 54 9E D6 25   7081

000400   0E 7F D4 E8 EB 21 3D D1   : 63
000408   A8 56 0F 9A 36 ED 8C 1F   : 75
000410   AD 09 01 3D D5 FB C9 3A   : C7
000418   62 E8 AC 18 3F F7 32 2F   : A5
000420   9D C5 DD 4F 60 6D A0 BE   : B9
000428   C4 DD 7C 32 85 53 56 FF   : 7C
000430   21 40 E3 77 2B 0F 82 FB   : 72
000438   CE 17 CC 43 AD C4 24 70   : F9
000440   79 90 2C AE 40 B7 BE F1   : 89
000448   99 54 71 D9 FE F1 F3 89   : A2
000450   1C 1A F0 D4 51 D2 7A 79   : 10
000458   5E 37 85 2B 7B 33 9E C1   : 52
000460   24 8B 24 FB DA A9 DC 9B   : C8
000468   6F 74 0D 7B EC 33 8B BF   : D4
000470   19 F6 5B 9E F5 B6 D2 B5   : 3A
000478   3A 15 6D EC 8D 45 64 EB   : C9
--------------------------------------
  SUM:   87 FE A3 98 44 17 C6 2F   1D52

000480   41 7D 9D D6 06 37 C1 3A   : 69
000488   4F 19 57 26 B0 62 7F 04   : 7A
000490   D5 7B F4 C4 D4 CB E5 AB   : 37
000498   31 C6 07 B9 4A 39 87 FC   : BD
0004A0   89 79 B5 D7 1F D6 60 C4   : A7
0004A8   FD 80 2E 7F 6A 20 03 F4   : AB
0004B0   EA AC 4E DE 0C B1 16 3A   : CF
0004B8   2D 5D 9A B1 85 CE B7 1E   : FD
0004C0   22 D7 EE A1 6D 37 0F BF   : FA
0004C8   28 D3 E4 8F CD 41 A3 77   : 96
0004D0   3E FC 56 89 1B AC 10 6C   : 5C
0004D8   E0 BE 90 0F 25 0A 96 53   : 55
0004E0   3D 2B AF 6C 54 BD 59 F0   : DD
0004E8   46 A3 B8 2D EF A7 95 1C   : 15
0004F0   3F F4 17 B3 56 0B B1 4F   : 5E
0004F8   B8 96 AE 17 15 8C F8 2B   : D7
--------------------------------------
  SUM:   15 95 9E 89 16 3B CB 70   42BB

000500   BB 52 0A 81 AD DC D0 5D   : 4E
000508   BA FB 01 1F 9B A2 A5 EF   : A6
000510   D2 C3 5F 61 58 61 6F 32   : AF
000518   63 72 50 D7 58 0D 55 D9   : 8F
000520   09 DD CC 96 DA 4A EC B5   : 0D
000528   AE 2A 9B 5F 25 BA B8 19   : 82
000530   A0 81 93 4D DD 1D 77 CE   : 40
000538   6D EE 2B B7 B1 CA 5B A9   : BC
000540   E3 C9 E6 2B B2 33 A4 05   : 4B
000548   9A B9 BD A8 8E 05 A9 FD   : F1
000550   60 37 0F F6 40 35 38 FE   : 47
000558   C5 B8 EA 7F E4 B4 D2 57   : A7
000560   40 E3 B9 5C 55 34 0E 4F   : 1E
000568   62 72 87 F4 18 15 E7 A5   : 08
000570   99 FF CB A0 A6 C6 CE 88   : C5
000578   BE 6A BC D4 64 47 17 EB   : 65
--------------------------------------
  SUM:   09 27 42 DD 60 4E E0 5A   1BBB

000580   86 DB 58 CF CD 0D 12 79   : ED
000588   A1 32 BD B7 B6 A2 4E 95   : 82
000590   48 06 74 72 E5 8C 53 C4   : BC
000598   0E 0A B2 96 B3 53 96 68   : 64
0005A0   6E 38 59 64 32 74 EA DA   : CD
0005A8   C5 61 7A 1A F4 80 28 37   : 8D
0005B0   2A 66 D4 EE 68 5B 2A 66   : A5
0005B8   76 30 34 72 A6 70 F7 E6   : 3F
0005C0   8A BA 2B BC F6 99 AB BC   : 21
0005C8   38 3E 1B 6E E8 63 0A 2D   : 81
0005D0   18 42 05 89 80 D4 8E 40   : 0A
0005D8   F3 17 D3 FB 4A 16 9D F2   : C7
0005E0   F2 A7 E8 1D 86 A2 1F 2E   : 13
0005E8   0F 74 2D 3B 0A 25 68 06   : 88
0005F0   22 F8 8C BD E3 2B B3 FF   : 23
0005F8   AE 5A A1 A5 33 EF D9 8F   : D8
--------------------------------------
  SUM:   EE 0A 76 D4 9D 14 6F 74   E472

000600   96 F1 97 8F 9B D8 BF 26   : 05
000608   EF 1C 2F 51 46 C2 8A 17   : 34
000610   C6 ED B1 5B 0F C4 DA F5   : 61
000618   44 8F AA A6 91 4C 72 D8   : 4A
000620   B4 E9 16 AD 67 A2 17 AF   : 2F
000628   F2 B3 FB 38 1B 06 75 51   : BF
000630   D9 36 1B EB B0 80 B2 53   : 4A
000638   87 A4 E1 7F 82 EE EE D0   : B9
000640   C6 2B B8 2F E5 FB 32 E2   : CC
000648   6D 49 0F D8 30 7E 21 EA   : 56
000650   DF D8 8F DE 57 E1 03 2A   : 89
000658   6E 05 70 FC 4A E6 CA 7F   : 58
000660   1D 88 3F 96 62 3C 0E FA   : 20
000668   84 3D 22 D0 5D A0 EF 91   : 30
000670   24 92 03 8A 98 1F 58 A7   : F9
000678   5D BC F3 15 1A 75 DF 1E   : AD
--------------------------------------
  SUM:   37 63 4B 16 5C 70 15 F2   0A76

000680   7A 2F EC 5D A1 67 52 32   : 7E
000688   0E 5D 79 35 7F 41 EB A6   : 6A
000690   DE 5A CC 7F 2F 18 A2 B9   : 25
000698   EF E3 17 33 A1 63 C5 47   : 2C
0006A0   6A 49 34 34 F0 2C B1 01   : E9
0006A8   43 FE C6 E1 FF 48 C7 FC   : F2
0006B0   12 9E 75 7D 92 3B 5E 0E   : DB
0006B8   93 89 87 B2 CB 97 20 9A   : 71
0006C0   BD 4B 3D 27 F2 F4 34 E2   : 68
0006C8   7F 73 22 70 3C 83 D6 4D   : 66
0006D0   94 7B AE DC E5 0F E9 F1   : 67
0006D8   8A 65 89 89 DC 69 24 F2   : 5C
0006E0   F8 AD D5 69 F9 6D 59 77   : 19
0006E8   39 C3 91 57 15 21 BE 04   : DC
0006F0   92 BE B4 63 E4 12 BD 97   : B1
0006F8   87 A9 D7 97 19 9C 82 78   : 4D
--------------------------------------
  SUM:   4B AC C5 3E 36 94 07 19   81D6

000700   7A 8A 23 76 03 74 68 47   : C3
000708   D7 97 17 6A C8 67 8C 7A   : 24
000710   4C BE 86 31 05 AF 9C 26   : 37
000718   B3 B5 7F 51 B8 E3 87 EE   : 48
000720   C9 BF 00 F9 F9 A3 46 E6   : 49
000728   86 65 E0 72 82 C5 D2 3B   : 91
000730   31 80 3B B3 65 5C 2A E1   : 6B
000738   AE B6 E0 31 56 2E F6 0B   : FA
000740   C2 1B 23 1D 25 D6 F9 BF   : D0
000748   D3 4B 43 61 1D 9C 03 F7   : 75
000750   CB 73 F4 C5 0B EA 59 AF   : F4
000758   3D 6A AB 93 36 F9 FA 77   : 85
000760   DB 43 88 1C 93 EC 91 D9   : AB
000768   DC 46 D8 99 1B 5A B5 48   : 05
000770   27 EB 89 8E 10 6E BB 73   : D5
000778   37 C2 4B F9 BA 10 25 E3   : 0F
--------------------------------------
  SUM:   30 67 73 C3 B9 78 C4 35   998F

000780   23 EC CB 8C FE 40 FC AA   : 4A
000788   E6 CA DF 3A FE 73 94 DD   : AB
000790   8B 03 77 6A EB F9 4E 53   : F4
000798   76 F5 4F 39 11 58 BB BC   : D3
0007A0   77 F7 5C 7A 4F FD 23 CE   : 81
0007A8   99 D6 46 74 6A 3B 32 9A   : 9A
0007B0   78 DB AB BE 5A 08 4C BC   : 26
0007B8   60 44 D6 9F B7 33 5B 2A   : 88
0007C0   EB DD 47 B4 86 9B 8F A9   : 1C
0007C8   97 DD FD AC A1 A2 9C 9D   : 99
0007D0   13 51 48 EC 86 E7 62 DB   : 42
0007D8   69 1F 80 D5 BA 7A 5F 1A   : 8A
0007E0   91 D2 70 FC FE 40 00 00   : 0D
0007E8   00 00 00 00 00 00 00 00   : 00
0007F0   00 00 00 00 00 00 00 00   : 00
0007F8   00 00 00 00 00 00 00 00   : 00
--------------------------------------
  SUM:   81 96 0F D1 27 55 81 1F   D4A4
```

▶祝！　荻窪圭さんご結婚。う～む，「アイドルのためのX68000」とか「女子大生のためのX68000」とかやってほしかったんですが，もう無理ですね。ともかくがんばってください（深読みはしないでね）。

今川　彰(24)東京都

リスト3

```
  1: *===================================
  2: *
  3: *            LHAFNC.S
  4: *
  5: * Programmed 1993 Mitsuru Kamiyama
  6: *===================================
  7:
  8:          .include        doscall.mac
  9:          .include        iocscall.mac
 10:          .include        fdef.h
 11:          .text
 12:          .even
 13:
 14:          dc.l    x_ret,x_ret,x_ret,x_ret
 15:          dc.l    x_ret,x_ret,x_ret,x_ret
 16:          dc.l    x_token
 17:          dc.l    x_param
 18:          dc.l    x_exec
 19:          dc.l    0,0,0,0,0
 20:
 21: x_ret:
 22:          rts
 23: x_token:
 24:          dc.b    'lha',0,0
 25:          .even
 26: x_param:
 27:          dc.l    lha_param
 28: lha_param:
 29:          dc.w    str_val
 30:          dc.w    aryl_fic
 31:          dc.w    int_ret
 32: x_exec:
 33:          dc.l    x_run
 34: x_run:
 35:          clr.l   ex_byte
 36:          clr.l   buf_byte
 37:          move.l  12(sp),a_file     *ファイル名のアドレス→a_file
 38:          move.l  22(sp),a0
 39:          move.l  a0,a_buf
 40:          addi.l  #10,a_buf         *BASIC側のバッファの先頭アドレス
 41:          moveq.l #0,d0
 42:          move.w  8(a0),d0          *型に合わせて最大バイト数を計算
 43:          addq.l  #1,d0
 44:          move.w  6(a0),d1
 45:          cmpi.w  #1,d1
 46:          beq     branch
 47:          cmpi.w  #4,d1
 48:          beq     shift
 49:          lsl.l   #1,d0             *FLOAT型
 50: shift:
 51:          lsl.l   #2,d0             *INT型
 52: branch:
 53:          move.l  d0,buf_byte
 54:          move.l  a7,sp_buf         *スタック(a7)を退避する
 55:          bra     lha               *LHAルーチンへブランチ(戻り値: d0=0 or 1)
 56: result_set:
 57:          movea.l sp_buf,a7         *スタックをブランチ前の状態に戻す
 58:          lea     ret_dat,a0        *戻り値をセット(d0=0の時有効)
 59:          lea     err_mes,a1        *エラーメッセージ(d0=1の時有効)
 60:          rts
 61:
 62:          .data
 63:
 64: ret_dat:                           *戻り値
 65:          dc.w    0
 66:          dc.l    0
 67: ex_byte:
 68:          dc.l    0                 *解凍バイト数
 69: buf_byte:
 70:          dc.l    0                 *バッファバイト数
 71: err_mes:
 72:          dc.b    'ファイルがオープンできないか、バッファが足りません。',0
 73:
 74:          .text
 75:          .even
 76:
 77: *
 78: *LHA解凍ルーチン (ORIGINAL: LHA's SFX 0.05 OKADA, Norio)
 79: *
 80:
 81: lha:
 82:          movem.l a0-a3,-(a7)
 83:          link    a6,#0
 84:          clr.w   -(a7)
 85:          move.l  a_file,-(sp)
 86:          DOS     _OPEN
 87:          addq.w  #6,a7
 88:          move.w  d0,fp
 89:          bge     L00128
 90:          bra     ng_exit
 91: L00128:
 92:          lea.l   L10c14,a4
 93:          moveq.l #$00,d7
 94: L0017a:
 95:          move.l  d7,d0
 96:          moveq.l #$07,d6
 97: L0017e:
 98:          lsr.w   #1,d0
 99:          bcc     L00186
100:          eori.w  #$a001,d0
101: L00186:
102:          dbra    d6,L0017e
103:          move.w  d0,(a4)+
104:          addq.b  #1,d7
105:          tst.b   d7
106:          bne     L0017a
107: L00192:
108:          lea.l   L10e15,a3
109:          move.w  fp,-(a7)
110:          DOS     _FGETC
111:          addq.w  #2,a7
112:          move.b  d0,(a3)+
113:          move.l  d0,d6
114:          beq     ok_exit
115:          addq.l  #1,d6
116:          move.l  d6,-(a7)
117:          move.l  a3,-(a7)
118:          move.w  fp,-(a7)
119:          DOS     _READ
120:          lea.l   $000a(a7),a7
121:          lea.l   L10e1c,a0
122:          move.b  $0003(a0),d0
123:          lsl.l   #8,d0
124:          move.b  $0002(a0),d0
125:          lsl.l   #8,d0
126:          move.b  $0001(a0),d0
127:          lsl.l   #8,d0
128:          move.b  (a0),d0
129:          move.l  d0,(a0)+
130:          move.b  $0003(a0),d0
131:          lsl.l   #8,d0
132:          move.b  $0002(a0),d0
133:          lsl.l   #8,d0
134:          move.b  $0001(a0),d0
135:          lsl.l   #8,d0
136:          move.b  (a0),d0
137:          move.l  d0,(a0)+
138:          move.b  $0001(a0),d0
139:          lsl.w   #8,d0
140:          move.b  (a0),d0
141:          move.w  d0,(a0)+
142:          move.b  $0001(a0),d0
143:          lsl.w   #8,d0
144:          move.b  (a0),d0
145:          move.w  d0,(a0)+
146:          move.b  (a3)+,d0
147:          subq.l  #1,d6
148: L0020a:
149:          sub.b   (a3)+,d0
150:          subq.l  #1,d6
151:          bne     L0020a
152:          tst.b   d0
153:          bne     L00242
154:          lea.l   L10e17,a3
155:          cmpi.b  #'-',(a3)+
156:          bne     L00242
157:          move.l  (a3)+,d0
158:          sub.l   #'lh0-',d0
159:          lsr.l   #8,d0
160:          cmpi.b  #'4',d0
161:          bne     L00232
162:          moveq.l #$00,d0
163: L00232:
164:          move.b  d0,L143a8
165:          beq     L0024c
166:          subq.l  #4,d0
167:          beq     L0024c
168:          subq.l  #1,d0
169:          beq     L0024c
170: L00242:
171:          bra     ng_exit
172: L0024c:
173:          lea.l   L10e2b,a5
174:          moveq.l #$00,d6
175:          move.b  -$0001(a5),d6
176:          adda.l  d6,a5
177:          move.b  $0001(a5),d5
178:          lsl.w   #8,d5
179:          move.b  (a5),d5
180:          move.w  d5,L10eb4
181:          clr.b   $0001(a5)
182:          clr.b   (a5)
183:          move.w  d5,d0
184:          bsr     L0097e
185:          move.b  #$20,(a5)+
186:          move.b  #$00,(a5)
187:          lea.l   L10e2b,a5
188:          move.b  #$00,-$0001(a5)
189: L002c6:
190:          tst.b   L143a9
191:          bne     L002d2
192:          bra     L00192
193: L002d2:
194:          moveq.l #$00,d0
195:          move.w  d0,L10eb6
196:          subq.l  #1,d0
197:          move.l  d0,L10eac
198:          cmpi.b  #$00,L143a8
199:          bne     L0030c
200:          bsr     L003f8
201:          bra     L00310
202: L0030c:
203:          bsr     L00442
204: L00310:
205:          move.w  L10eb6,d0
206:          cmp.w   L10eb4,d0
207:          moveq.l #$00,d6
208:          move.b  L10e28,d6
209:          bra     L00192
210: ng_exit:                                   *エラーにより終了(d0=1)
211:          moveq.l #1,d0
212:          bra     exit
213: ok_exit:                                   *正常終了(d0=0)
214:          moveq.l #0,d0
215: exit:
216:          move.w  fp,d5
217:          bsr.w   close
218:          unlk    a6
219:          movem.l (a7)+,a0-a3
220:          jmp     result_set
221: L003a4:
222:          moveq.l #$59,d0
223:          cmpi.b  #$59,d0
224:          rts
225: close:
226:          tst.w   d5
227:          bmi     L003dc
228:          movem.l d0,-(sp)
229:          move.w  d5,-(a7)
230:          DOS     _CLOSE
231:          addq.w  #2,a7
232:          movem.l (sp)+,d0
233: L003dc:
234:          rts
235: L003f8:
236:          move.l  #$00008000,d6
237:          cmp.l   L10e20,d6
238:          bls     L0040c
239:          move.l  L10e20,d6
240: L0040c:
241:          move.l  d6,-(a7)
242:          pea.l   buf1
243:          move.w  fp,-(a7)
244:          DOS     _READ
245:          lea.l   $000a(a7),a7
246:          sub.l   d6,L10e20
247:          cmp.l   d0,d6
248:          beq     L00434
249:          bra     ng_exit
250: L00434:
251:          bsr     kakidashi
252:          tst.l   L10e20
253:          bgt     L003f8
254:          rts
```

▶ 2月号の清水さんへ。あの人は「ながしま」さんです。テレビに出演したこともあります。ちなみにメッ○サン○ーの2階の売り場では，「ながしま」というCGが「いかがっすか〜」とお客に呼びかけています(いまはないかも)。一度そのCGを見るとくせになってしまいますよ。フフフ……。

大内　章光(21)東京都

```
255: L00442:
256:          movem.l d3-d5/a3-a4,-(a7)
257:          moveq.l #$00,d0
258:          move.w  d0,L14324
259:          move.b  d0,L14326
260:          move.b  d0,L14327
261:          move.w  d0,L10eb8
262:          lea.l   buf1,a3
263:          movea.l a3,a4
264:          move.l  L10e20,d5
265:          move.l  #$00000010,-(a7)
266:          bsr     L00918
267:          addq.w  #4,a7
268: L0047a:
269:          tst.l   d5
270:          bls     L004e2
271:          bsr     L00526
272:          cmpi.w  #$00ff,d0
273:          bhi     L004a4
274:          move.b  d0,(a3)+
275:          cmpa.l  #buf2,a3
276:          bne     L004a0
277:          move.l  a3,-(a7)
278:          bsr     L0083e
279:          addq.w  #4,a7
280:          lea.l   buf1,a3
281: L004a0:
282:          subq.l  #1,d5
283:          bra     L0047a
284: L004a4:
285:          move.l  d0,d3
286:          subi.l  #$000000fd,d3
287:          sub.l   d3,d5
288:          bsr     L005be
289:          move.l  a3,d4
290:          sub.l   a4,d4
291:          sub.l   d0,d4
292:          subq.l  #1,d4
293:          subq.l  #1,d3
294: L004bc:
295:          andi.w  #$7fff,d4
296:          move.b  $00(a4,d4.w),(a3)+
297:          cmpa.l  #buf2,a3
298:          bne     L004da
299:          move.l  a3,-(a7)
300:          bsr     L0083e
301:          addq.w  #4,a7
302:          lea.l   buf1,a3
303: L004da:
304:          addq.w  #1,d4
305:          dbra    d3,L004bc
306:          bra     L0047a
307: L004e2:
308:          move.l  a3,-(a7)
309:          bsr     L0083e
310:          addq.w  #4,a7
311:          movem.l (a7)+,d3-d5/a3-a4
312:          rts
313: L004f0:
314:          link    a6,#0
315:          movem.l d1/a3-a4,-(a7)
316:          move.l  $0008(a6),d1
317:          lea.l   L116b0,a3
318:          lea.l   L10eba,a4
319: L00508:
320:          cmp.l   d1,d0
321:          bcs     L0051e
322:          lsl.w   #1,d0
323:          lsl.b   #1,d3
324:          bcc     L00518
325:          move.w  $00(a3,d0.w),d0
326:          bra     L00508
327: L00518:
328:          move.w  $00(a4,d0.w),d0
329:          bra     L00508
330: L0051e:
331:          movem.l (a7)+,d1/a3-a4
332:          unlk    a6
333:          rts
334: L00526:
335:          move.l  d3,-(a7)
336:          subq.w  #1,L10eb8
337:          bcc     L0057c
338:          move.l  #$00000010,-(a7)
339:          bsr     L008f0
340:          addq.w  #4,a7
341:          subq.w  #1,d0
342:          move.w  d0,L10eb8
343:          move.l  #$00000003,-(a7)
344:          move.l  #$00000005,-(a7)
345:          move.l  #$00000013,-(a7)
346:          bsr     L00742
347:          lea.l   $000c(a7),a7
348:          bsr     L00618
349:          move.l  #$ffffffff,-(a7)
350:          move.l  #$00000004,-(a7)
351:          move.l  #$0000000e,-(a7)
352:          bsr     L00742
353:          lea.l   $000c(a7),a7
354: L0057c:
355:          lea.l   L11ea6,a0
356:          moveq.l #$00,d0
357:          move.w  L14324,d0
358:          move.l  d0,d3
359:          lsl.w   #4,d3
360:          lsr.w   #4,d0
361:          lsl.w   #1,d0
362:          move.w  $00(a0,d0.w),d0
363:          move.l  #$000001fe,-(a7)
364:          bsr     L004f0
365:          addq.w  #4,a7
366:          lea.l   L140a6,a0
367:          moveq.l #$00,d1
368:          move.b  $00(a0,d0.w),d1
369:          move.l  d0,-(a7)
370:          move.l  d1,-(a7)
371:          bsr     L00918
372:          addq.w  #4,a7
373:          move.l  (a7)+,d0
374:          move.l  (a7)+,d3
375:          rts
376: L005be:
377:          move.l  d3,-(a7)
378:          moveq.l #$00,d0
379:          move.w  L14324,d0
380:          move.l  d0,d3
381:          lsr.w   #8,d0
382:          add.l   d0,d0
383:          lea.l   L13ea6,a0
384:          move.w  $00(a0,d0.l),d0
385:          move.l  #$0000000e,-(a7)
386:          bsr     L004f0
387:          addq.w  #4,a7
388:          lea.l   L142a4,a0
389:          moveq.l #$00,d1
390:          move.b  $00(a0,d0.w),d1
391:          move.l  d0,-(a7)
392:          move.l  d1,-(a7)
393:          bsr     L00918
394:          addq.w  #4,a7
395:          move.l  (a7)+,d0
396:          cmpi.w  #$0001,d0
397:          bls     L00614
398:          subq.l  #1,d0
399:          move.l  d0,d3
400:          move.l  d0,-(a7)
401:          bsr     L008f0
402:          addq.w  #4,a7
403:          moveq.l #$01,d2
404:          lsl.l   d3,d2
405:          add.l   d2,d0
406: L00614:
407:          move.l  (a7)+,d3
408:          rts
409: L00618:
410:          movem.l d3-d7/a3-a5,-(a7)
411:          lea.l   L140a6,a3
412:          move.l  #$00000009,-(a7)
413:          bsr     L008f0
414:          addq.w  #4,a7
415:          cmpi.w  #$01fe,d0
416:          bhi     ng_exit
417:          tst.w   d0
418:          bne     L00674
419:          move.l  #$000001fe,d1
420:          tst.l   d1
421:          ble     L0064c
422:          subq.l  #1,d1
423: L00646:
424:          move.b  d0,(a3)+
425:          dbra    d1,L00646
426: L0064c:
427:          move.l  #$00000009,-(a7)
428:          bsr     L008f0
429:          addq.w  #4,a7
430:          move.l  #$00001000,d1
431:          lea.l   L11ea6,a0
432:          tst.l   d1
433:          ble     L00670
434:          subq.l  #1,d1
435: L0066a:
436:          move.w  d0,(a0)+
437:          dbra    d1,L0066a
438: L00670:
439:          bra     L0073c
440: L00674:
441:          movea.l a3,a5
442:          adda.l  d0,a5
443: L00678:
444:          cmpa.l  a5,a3
445:          bge     L00704
446:          moveq.l #$00,d0
447:          move.w  L14324,d0
448:          move.l  d0,d3
449:          lsr.w   #8,d0
450:          lsl.w   #1,d0
451:          lea.l   L13ea6,a0
452:          move.w  $00(a0,d0.w),d0
453:          move.l  #$00000013,-(a7)
454:          bsr     L004f0
455:          addq.w  #4,a7
456:          move.l  d0,d3
457:          moveq.l #$00,d1
458:          lea.l   L142a4,a0
459:          move.b  $00(a0,d0.w),d1
460:          move.l  d1,-(a7)
461:          bsr     L00918
462:          addq.w  #4,a7
463:          subq.l  #2,d3
464:          bhi     L006fe
465:          bne     L006d2
466:          move.l  #$00000009,-(a7)
467:          bsr     L008f0
468:          addq.w  #4,a7
469:          moveq.l #$14,d3
470:          add.l   d0,d3
471:          bra.w   L006ec
472: L006d2:
473:          addq.l  #1,d3
474:          bne     L006ea
475:          move.l  #$00000004,-(a7)
476:          bsr     L008f0
477:          addq.w  #4,a7
478:          addq.l  #3,d0
479:          move.l  d0,d3
480:          bra.w   L006ec
481: L006ea:
482:          moveq.l #$01,d3
483: L006ec:
484:          moveq.l #$00,d0
485:          tst.l   d3
486:          ble     L006fa
487:          subq.l  #1,d3
488: L006f4:
489:          move.b  d0,(a3)+
490:          dbra    d3,L006f4
491: L006fa:
492:          bra     L00678
493: L006fe:
494:          move.b  d3,(a3)+
495:          bra     L00678
496: L00704:
497:          moveq.l #$00,d0
498:          move.l  #L142a4,d1
499:          sub.l   a3,d1
500:          tst.l   d1
501:          ble     L0071a
502:          subq.l  #1,d1
503: L00714:
504:          move.b  d0,(a3)+
505:          dbra    d1,L00714
506: L0071a:
507:          pea.l   L11ea6
508:          move.l  #$0000000c,-(a7)
509:          pea.l   L140a6
510:          move.l  #$000001fe,-(a7)
511:          jsr     L009d6
512:          lea.l   $0010(a7),a7
```

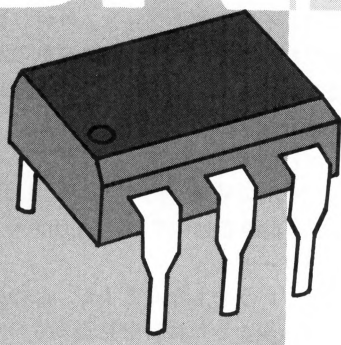

▶私の化学の単位は「新年，教授に勝ったら＋15点ボウリング大会」にかかっていたが，年のわりに教授は強く，あえなく撃沈された。　角谷　光憲(19)愛知県

```
513: L0073c:
514:            movem.l (a7)+,d3-d7/a3-a5
515:            rts
516: L00742:
517:            link    a6,#0
518:            movem.l a3-a5,-(a7)
519:            move.l  $000c(a6),-(a7)
520:            bsr     L008f0
521:            addq.w  #4,a7
522:            cmp.l   $0008(a6),d0
523:            bhi     ng_exit
524:            lea.l   L142a4,a3
525:            tst.w   d0
526:            bne     L0079c
527:            move.l  $0008(a6),d1
528:            tst.l   d1
529:            ble     L00776
530:            subq.l  #1,d1
531: L00770:
532:            move.b  d0,(a3)+
533:            dbra    d1,L00770
534: L00776:
535:            move.l  $000c(a6),-(a7)
536:            bsr     L008f0
537:            addq.w  #4,a7
538:            lea.l   L13ea6,a0
539:            move.l  #$00000100,d1
540:            tst.l   d1
541:            ble     L00798
542:            subq.l  #1,d1
543: L00792:
544:            move.w  d0,(a0)+
545:            dbra    d1,L00792
546: L00798:
547:            bra     L00836
548: L0079c:
549:            movea.l a3,a4
550:            adda.l  d0,a4
551:            movea.l a3,a5
552:            adda.l  $0010(a6),a5
553: L007a6:
554:            cmpa.l  a4,a3
555:            bcc.w   L007fc
556:            move.l  #$00000003,-(a7)
557:            bsr     L008f0
558:            addq.w  #4,a7
559:            cmpi.b  #$07,d0
560:            bne     L007da
561:            move.w  L14324,d1
562: L007c4:
563:            lsl.w   #1,d1
564:            bcc     L007cc
565:            addq.w  #1,d0
566:            bra     L007c4
567: L007cc:
568:            move.l  d0,-(a7)
569:            subq.l  #6,d0
570:            move.l  d0,-(a7)
571:            bsr     L00918
572:            addq.w  #4,a7
573:            move.l  (a7)+,d0
574: L007da:
575:            move.b  d0,(a3)+
576:            cmpa.l  a5,a3
577:            bne     L007fa
578:            move.l  #$00000002,-(a7)
579:            bsr     L008f0
580:            addq.w  #4,a7
581:            moveq.l #$00,d1
582:            tst.l   d0
583:            ble     L007fa
584:            subq.l  #1,d0
585: L007f4:
586:            move.b  d1,(a3)+
587:            dbra    d0,L007f4
588: L007fa:
589:            bra     L007a6
590: L007fc:
591:            move.l  #L142a4,d1
592:            add.l   $0008(a6),d1
593:            sub.l   a3,d1
594:            moveq.l #$00,d0
595:            tst.l   d1
596:            ble     L00816
597:            subq.l  #1,d1
598: L00810:
599:            move.b  d0,(a3)+
600:            dbra    d1,L00810
601: L00816:
602:            pea.l   L13ea6
603:            move.l  #$00000008,-(a7)
604:            pea.l   L142a4
605:            move.l  $0008(a6),-(a7)
606:            jsr     L009d6
607:            lea.l   $0010(a7),a7
608: L00836:
609:            movem.l (a7)+,a3-a5
610:            unlk    a6
611:            rts
612: L0083e:
613:            link    a6,#0
614:            move.l  d6,-(a7)
615:            move.l  $0008(a6),d6
616:            sub.l   #buf1,d6
617:            bsr     kakidashi
618:            move.l  (a7)+,d6
619:            unlk    a6
620:            rts
621: kakidashi:
622:            tst.l   d6
623:            beq     L008aa
624:
625:            movem.l d6,-(sp)            *ここから、バッファ書き出し部
626:            move.l  ex_byte,d1
627:            add.l   d6,d1
628:            move.l  d1,ex_byte
629:            cmp.l   buf_byte,d1
630:            bgt     ng_exit             *BASIC側のバッファが足りない
631:            movea.l a_buf,a1
632:            lea.l   buf1,a0
633:            subq.l  #1,d6
634: loop:                                   *BASIC側のバッファに転送
635:            move.b  (a0)+,(a1)+
636:            dbra    d6,loop
637:            move.l  a1,a_buf
638:            movem.l (sp)+,d6            *ここまで
639:
640:            lea.l   buf1,a0
641:            lea.l   L10c14,a1
642:            moveq.l #$00,d1
643:            move.w  L10eb6,d1
644: L00888:
645:            moveq.l #$00,d0
646:            move.b  (a0)+,d0
647:            eor.b   d1,d0
648:            lsl.w   #1,d0
649:            move.w  $00(a1,d0.w),d0
650:            lsr.w   #8,d1
651:            eor.w   d0,d1
652:            subq.l  #1,d6
653:            bne     L00888
654:            move.w  d1,L10eb6
655: L008aa:
656:            rts
657: L008ac:
658:            movea.l L10eac,a0
659:            cmpa.l  #L10c14,a0
660:            bcs     L008e4
661:            move.l  #$00008000,d0
662:            sub.l   d0,L10e1c
663:            bcc     L008ce
664:            add.l   L10e1c,d0
665: L008ce:
666:            lea.l   buf2,a0
667:            move.l  d0,-(a7)
668:            move.l  a0,-(a7)
669:            move.w  fp,-(a7)
670:            DOS     _READ
671:            lea.l   $000a(a7),a7
672: L008e4:
673:            moveq.l #$00,d0
674:            move.b  (a0)+,d0
675:            move.l  a0,L10eac
676:            rts
677: L008f0:
678:            link    a6,#0
679:            move.l  d3,-(a7)
680:            moveq.l #$10,d0
681:            sub.l   $0008(a6),d0
682:            moveq.l #$00,d3
683:            move.w  L14324,d3
684:            lsr.w   d0,d3
685:            move.l  $0008(a6),-(a7)
686:            bsr.w   L00918
687:            addq.w  #4,a7
688:            move.l  d3,d0
689:            move.l  (a7)+,d3
690:            unlk    a6
691:            rts
692: L00918:
693:            link    a6,#0
694:            movem.l d4-d7,-(a7)
695:            moveq.l #$00,d0
696:            move.b  L14326,d0
697:            moveq.l #$00,d4
698:            move.w  L14324,d4
699:            moveq.l #$00,d5
700:            move.b  L14327,d5
701:            move.l  $0008(a6),d6
702:            cmp.b   d5,d6
703:            ble     L0095a
704:            sub.b   d5,d6
705:            lsl.w   d5,d4
706:            rol.b   d5,d0
707:            add.b   d0,d4
708:            moveq.l #$08,d5
709: L0094a:
710:            bsr     L008ac
711:            cmp.b   d5,d6
712:            ble     L0095a
713:            sub.b   d5,d6
714:            lsl.w   d5,d4
715:            move.b  d0,d4
716:            bra     L0094a
717: L0095a:
718:            sub.b   d6,d5
719:            move.b  d5,L14327
720:            lsl.w   d6,d0
721:            move.b  d0,L14326
722:            lsr.w   #8,d0
723:            lsl.w   d6,d4
724:            add.b   d0,d4
725:            move.w  d4,L14324
726:            movem.l (a7)+,d4-d7
727:            unlk    a6
728:            rts
729: L0097e:
730:            lea.l   L10e2b,a3
731: L00984:
732:            move.b  (a3)+,d0
733:            cmpi.b  #$20,d0
734:            bls     L009a2
735:            cmpi.b  #$2f,d0
736:            beq     L00998
737:            cmpi.b  #$5c,d0
738:            bne     L00984
739: L00998:
740:            move.b  #$5c,-$0001(a3)
741:            move.l  a3,d6
742:            bra     L00984
743: L009a2:
744:            clr.b   -(a3)
745:            move.b  -(a3),d0
746:            subi.b  #$5c,d0
747:            move.b  d0,L143a9
748:            lea.l   L10e2b,a3
749: L009b6:
750:            move.b  (a3),d0
751:            cmpi.b  #$5c,d0
752:            bne     L009ce
753:            move.b  #$5c,(a3)
754: L009ce:
755:            addq.l  #1,a3
756:            cmpa.l  d6,a3
757:            bcs     L009b6
758:            rts
759: L009d6:
760:            link    a6,#-12
761:            movem.l d3-d7/a3-a5,-(a7)
762:            movea.l $0010(a6),a4
763:            moveq.l #$01,d2
764:            lea.l   L143f0,a0
765:            addq.w  #2,a0
766:            moveq.l #$10,d5
767: L009ee:
768:            clr.w   (a0)+
769:            addq.l  #1,d2
770:            cmp.l   d2,d5
```

▶Oh!FMのときもそうでしたが，Oh!Xの雰囲気に慣れるのに６カ月もかかってしまった。
これがOh!シリーズの特色なのか，それとも自分に順応力が足りないのだろうか。
服部　広一郎(19)宮城県

```
771:            bcc     L009ee
772:            moveq.l #$00,d2
773:            cmp.l   $0008(a6),d2
774:            bcc     L00a1e
775:            moveq.l #$00,d1
776:            lea.l   L143f0,a0
777: L00a06:
778:            movea.l $000c(a6),a5
779:            move.b  $00(a5,d2.l),d1
780:            move.l  d1,d0
781:            add.l   d0,d0
782:            addq.w  #1,$00(a0,d0.l)
783:            addq.l  #1,d2
784:            cmp.l   $0008(a6),d2
785:            bcs     L00a06
786: L00a1e:
787:            lea.l   L143aa,a0
788:            addq.w  #2,a0
789:            clr.w   (a0)
790:            moveq.l #$01,d2
791:            moveq.l #$00,d1
792:            lea.l   L143ac,a2
793:            addq.w  #2,a2
794:            lea.l   L143aa,a1
795:            addq.w  #2,a1
796:            lea.l   L143f0,a0
797:            addq.w  #2,a0
798: L00a44:
799:            move.w  (a0)+,d1
800:            moveq.l #$10,d0
801:            sub.l   d2,d0
802:            move.l  d1,d5
803:            asl.l   d0,d5
804:            move.l  d5,d0
805:            add.w   (a1)+,d0
806:            move.w  d0,(a2)+
807:            addq.l  #1,d2
808:            moveq.l #$10,d5
809:            cmp.l   d2,d5
810:            bcc     L00a44
811:            moveq.l #$10,d5
812:            sub.l   a4,d5
813:            move.l  d5,-$0008(a6)
814:            moveq.l #$01,d2
815:            cmpa.l  d2,a4
816:            bcs     L00a98
817:            moveq.l #$00,d3
818:            lea.l   L143ce,a1
819:            addq.w  #2,a1
820:            lea.l   L143aa,a0
821:            addq.w  #2,a0
822: L00a7c:
823:            move.w  (a0),d3
824:            move.l  d3,d0
825:            move.l  -$0008(a6),d5
826:            asr.l   d5,d0
827:            move.w  d0,(a0)+
828:            move.l  a4,d0
829:            sub.l   d2,d0
830:            moveq.l #$01,d1
831:            lsl.l   d0,d1
832:            move.w  d1,(a1)+
833:            addq.l  #1,d2
834:            cmpa.l  d2,a4
835:            bcc     L00a7c
836: L00a98:
837:            moveq.l #$10,d5
838:            cmp.l   d2,d5
839:            bcs     L00abc
840:            move.l  d2,d0
841:            add.l   d0,d0
842:            movea.l d0,a0
843:            adda.l  #L143ce,a0
844:            moveq.l #$10,d5
845: L00aac:
846:            moveq.l #$10,d0
847:            sub.l   d2,d0
848:            moveq.l #$01,d1
849:            lsl.l   d0,d1
850:            move.w  d1,(a0)+
851:            addq.l  #1,d2
852:            cmp.l   d2,d5
853:            bcc     L00aac
854: L00abc:
855:            lea.l   L143ac,a0
856:            move.l  a4,d1
857:            add.l   d1,d1
858:            moveq.l #$00,d0
859:            move.w  $00(a0,d1.l),d0
860:            move.l  d0,d2
861:            move.l  -$0008(a6),d5
862:            asr.l   d5,d2
863:            beq     L00af6
864:            moveq.l #$01,d1
865:            move.l  a4,d5
866:            lsl.l   d5,d1
867:            cmp.l   d2,d1
868:            beq     L00af6
869:            moveq.l #$01,d1
870:            lsl.l   d5,d1
871:            move.l  d2,d0
872:            add.l   d0,d0
873:            movea.l d0,a0
874:            adda.l  $0014(a6),a0
875: L00aee:
876:            clr.w   (a0)+
877:            addq.l  #1,d2
878:            cmp.l   d2,d1
879:            bne     L00aee
880: L00af6:
881:            move.l  $0008(a6),d7
882:            moveq.l #$0f,d0
883:            sub.l   a4,d0
884:            moveq.l #$01,d5
885:            move.l  d5,-$0004(a6)
886:            move.l  -$0004(a6),d5
887:            lsl.l   d0,d5
888:            move.l  d5,-$0004(a6)
889:            moveq.l #$00,d6
890:            cmp.l   d6,d7
891:            bls     L00c0a
892:            moveq.l #$00,d4
893: L00b18:
894:            moveq.l #$00,d0
895:            movea.l $000c(a6),a5
896:            move.b  $00(a5,d6.l),d0
897:            move.l  d0,-$000c(a6)
898:            beq     L00c00
899:            add.l   d0,d0
900:            lea.l   L143aa,a5
901:            move.w  $00(a5,d0.l),d4
902:            lea.l   L143ce,a0
903:            moveq.l #$00,d1
904:            move.w  $00(a0,d0.l),d1
905:            movea.l d4,a3
906:            adda.l  d1,a3
907:            cmpa.l  -$000c(a6),a4
908:            bcs     L00b6c
909:            move.l  d4,d2
910:            cmpa.l  d2,a3
911:            bls     L00bf0
912:            move.w  d6,d1
913:            move.l  d4,d0
914:            add.l   d0,d0
915:            movea.l d0,a0
916:            adda.l  $0014(a6),a0
917: L00b60:
918:            move.w  d1,(a0)+
919:            addq.l  #1,d2
920:            cmpa.l  d2,a3
921:            bhi     L00b60
922:            bra     L00bf0
923: L00b6c:
924:            move.l  -$000c(a6),d0
925:            add.l   d0,d0
926:            lea.l   L143aa,a5
927:            move.w  $00(a5,d0.l),d4
928:            move.l  d4,d3
929:            move.l  d3,d0
930:            move.l  -$0008(a6),d5
931:            lsr.l   d5,d0
932:            add.l   d0,d0
933:            movea.l $0014(a6),a0
934:            adda.l  d0,a0
935:            move.l  -$000c(a6),d2
936:            sub.l   a4,d2
937:            subq.l  #1,d2
938:            moveq.l #$ff,d5
939:            cmp.l   d2,d5
940:            beq     L00bee
941:            moveq.l #$00,d1
942:            move.l  d7,d0
943:            add.l   d0,d0
944:            movea.l d0,a2
945:            adda.l  #L116b0,a2
946:            move.l  d7,d0
947:            add.l   d0,d0
948:            movea.l d0,a1
949:            adda.l  #L10eba,a1
950: L00bb6:
951:            tst.w   (a0)
952:            bne     L00bc2
953:            clr.w   (a1)+
954:            clr.w   (a2)+
955:            move.w  d7,(a0)
956:            addq.l  #1,d7
957: L00bc2:
958:            move.l  d3,d0
959:            and.l   -$0004(a6),d0
960:            beq     L00bd8
961:            move.w  (a0),d1
962:            move.l  d1,d0
963:            add.l   d0,d0
964:            lea.l   L116b0,a0
965:            bra     L00be4
966: L00bd8:
967:            move.w  (a0),d1
968:            move.l  d1,d0
969:            add.l   d0,d0
970:            lea.l   L10eba,a0
971: L00be4:
972:            adda.l  d0,a0
973:            lsl.l   #1,d3
974:            dbra    d2,L00bb6
975:            ext.l   d2
976: L00bee:
977:            move.w  d6,(a0)
978: L00bf0:
979:            move.l  -$000c(a6),d0
980:            add.l   d0,d0
981:            lea.l   L143aa,a5
982:            move.w  a3,$00(a5,d0.l)
983: L00c00:
984:            addq.l  #1,d6
985:            cmp.l   $0008(a6),d6
986:            bcs     L00b18
987: L00c0a:
988:            movem.l -$002c(a6),d3-d7/a3-a5
989:            unlk    a6
990:            rts
991:
992:            .even
993:            .bss
994:
995: fp:     ds.w    1                       *ファイルポインタ
996: sp_buf: ds.l    1                       *スタック退避
997: a_buf:  ds.l    1                       *引数バッファのアドレス
998: a_file: ds.l    1                       *引数ファイル名のアドレス
999: buf1:   ds.b    32768
1000: buf2:  ds.b    32768
1001: L10c14: ds.b   513
1002: L10e15: ds.b   2
1003: L10e17: ds.b   5
1004: L10e1c: ds.l   1
1005: L10e20: ds.l   1
1006: L10e24: ds.w   1
1007: L10e26: ds.w   1
1008: L10e28: ds.b   3
1009: L10e2b: ds.b   129
1010: L10eac: ds.l   1
1011: L10eb4: ds.w   1
1012: L10eb6: ds.w   1
1013: L10eb8: ds.w   1
1014: L10eba: ds.b   2038
1015: L116b0: ds.b   2038
1016: L11ea6: ds.b   8192
1017: L13ea6: ds.b   512
1018: L140a6: ds.b   510
1019: L142a4: ds.b   128
1020: L14324: ds.w   1
1021: L14326: ds.b   1
1022: L14327: ds.b   129
1023: L143a8: ds.b   1
1024: L143a9: ds.b   1
1025: L143aa: ds.b   2
1026: L143ac: ds.b   34
1027: L143ce: ds.b   34
1028: L143f0: ds.b   34
1029:
1030:           .end
```

ONE CHIP IC

ワンチップIC工作入門（第2回）

ノイズリダクションを作る

Takao Katsuhiko
高尾 克彦

不定期連載として始まることになった「1チップIC工作入門」です。今回も音楽関係で，音声信号のノイズ低減装置を作ります。エコーとあわせて，やがてはカラオケシステムという野望もチラホラ……。

その昔，「巨人の星」というマンガに花形満というキャラクターがいました。主人公，星飛雄馬のライバルのひとりで，中学生のくせにスポーツカーを乗り回していたり，右を向いても左を向いても髪型が変わらなかったりと，なかなか不思議なキャラクターでした。彼は含蓄あるいくつかの言葉を残すのですが，そのなかに，
「白鳥の泳ぎは，一見優雅だが，水面下では……云々」

という文句がありました。私はてっきり，どこかの諺だと思っていたのですが，この花形満がオリジナルだそうです。

なんだかんだいっても，X68000はかなりそこらへんを意識していて，この水面下で起こっていることをあまりユーザーに意識させません。逆に，この正反対にいるのが，DOS/Vマシンで，これはもう「ロッキー」のスタローンのように「俺はこんなに頑張っているんだぞぅ」と雄叫びながら動作するコンピュータです。このようなジャジャ馬を，なだめすかしながらコンピュータを使っていくのもまた一興なのですが，ロデオをやりながら馬車は引けません。

さて，DOS/Vマシンのように，ハードウェアがかなり前面に押し出されている機械は別として，うまくソフィティスケイトされている機械では，ハードウェアの知識はほとんど要求されません。X68000のケースをドライバーで開け，内部をのぞいたことがある人はほとんどいないでしょう。

しかし，ドライバーでケースを開けなくても，おやっ？　と思うところがあります。X68000の前面にあるオーディオ端子です。背面のオーディオ端子には，そんなにノイズが乗っていないようですので，やはりフロッピードライブの下を信号が通り抜けてくるときにノイズをもらってきてしまうのでしょう。特に，ディスクを回したりするとそのメカノイズ(?)がもろに信号に乗ってしまいます。

ノイズのこと

ノイズの発生に関して，理由はいろいろ考えられます。

たとえば，フロッピードライブを回転させるときに，一時的にそちらのほうへ電力が取られて，ステレオ信号用の電位が下がってしまうとか（ほら，電子レンジや冷房のスイッチを入れると，一瞬部屋の灯りが暗くなるでしょ）。

また，モーターというのはコイルに電気を流したり，流さなかったりして，磁界を制御するものですから，モーターから漏れた磁界は，いろいろと電気的なノイズを発生します。それから，真っ直ぐに伸ばしたコードとラジカセのアンテナは電気的に同じものです。そういうわけで，ラジオの放送局と受信機のように，オーディオ信号にモーターの発生するノイズが乗るのかもしれません。

X68000の場合，そのノイズが嫌ならば，ノイズが乗る前の信号を背面のステレオジャックから引っ張ってくればよいのですが，そのような，ノイズ対策がとれないときもあるでしょう。

MIDI楽器の場合とか，システムの配線の都合とか，部屋のレイアウトだとか，どうしてもノイズの発生源の近くにコードを通さなければいけないときもあるでしょう。台湾製のIBM互換機をMIDI信号の制御に使っている場合もあるかもしれません。

また，精度を倍にしようと思ったら，価格のほうは10倍になってしまうというような，精度と価格の厳しい関係がアナログ回路にはあります。

もっとも，前回のエコー装置のように，
「どうせ，エフェクタというのは一種の音を壊す機械なんだから」

と，思いっきりノイズ対策をさぼってしまって，あとから，しまった，となることもあるかもしれません。

そのようなときに，ノイズ源をなくすのではなく，なんとかして信号からノイズを取り除こうというのが，今回製作するノイズリダクションです。

ノイズリダクションの種類

ノイズリダクションには，大きく分けて2つの種類があります。コンプリメンタル型（相補型）と，そうでないノンコンプリメンタル型（非相補型）です。

コンプリメンタルというのは，

complete（形，完全な）

↓

complement（他動，～を補って完全にする）

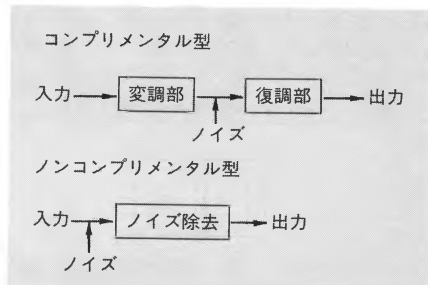
↓

complementary（形，補充の）

という言葉みたいです。発音記号を見ると，「カンプリメンタリ」と読むみたいですが，まぁいいでしょう。

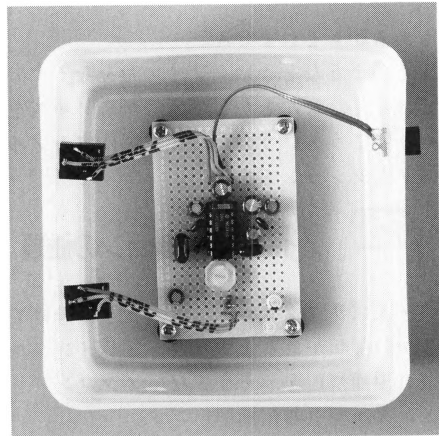
なにを補うかというと，ノイズリダクションの構成を2つに分けて，それらがお互いに協力して動作するようにするのです。

信号がノイズを受ける場所に入る前に，このノイズリダクションその1（変調部）が信号に細工を加えてやるのです。信号がノイズをもらいやすいところを通過したあとで，ノイズリダクションその2（復調部）のほうが，先ほどの細工を元に戻します（図

図1　ノイズリダクションの種類

1)。

皆さんもよく知っている，ドルビーNRシステムというのはこの方式です（詳しくは囲みを参照してください）。

この方式はノイズに弱いところをシステムの変調部/復調部ではさんでやるというものです。装置が2ついるので大変そうです。また，どこでノイズを拾ってしまうのか，そのノイズにはどんな性質があるのか，をはっきりと知っていなければなりません。

残りのノンコンプリメンタル型というのは，信号の細工をしないで，出力信号を監視して，その中にノイズが含まれていれば，除去しようというものです。

完成した基板

こうやって書くと，なにやら凄そうな感じがしますが，実際には，ある一定の電圧に満たない小さな信号はすべてノイズであるとみなして通さない，とか，その程度です。

この方式は，ノイズの除去率はあまり上がらない代わりに，信号の出力部に取り付けてやるだけですから，システムがわりあい手軽に構築できます。また，出力信号にどのような信号が混ざるとノイズであるとみなすのかを決めておけばよいだけですから，どこで，どのようなノイズが，というような解析をしないで，アバウトな感覚で使えるのが今回の製作（と私の性格）にピッタリの方法です。

そういうわけで，今回製作するのは，このノンコンプリメンタル型のノイズリダクションです。

LM1894というLSI

さて，今月使うLSIはNational Semiconductor社のLM1894という奴です。データシートをそのまま引用すると（一応，和訳したのは私です），「ノンコンプリメンタルなタイプのノイズリダクション。外付け部品に高価なものがなく，また回路定数の調整も必要ない。すでにあるテープやFM放送に対して有効。CCIR/ARM(なんじゃこりゃ）をかけたテープのノイズリダクションに対し，10dBの効果が期待できる。4.5V～18Vまでの電源で動作する。1Vrms入力でオーバーロード」。

で，自分でいうのもなんですが，私の英単語の語彙の少なさには目を見張るものがありますが，オーディオ音痴ぶりも相当なもんです。恥の上塗りをする前に，話を先に進めましょう。

理論

なにをノイズとするかという問題は非常に難しいものです。あるしきい値を決めてその値に満たないものをノイズと決めたり，扱う音域をあらかじめ決めておいてそれ以外をノイズとしたり……。

これらのノイズの定義の数だけ，ノイズリダクションの方式もあります（と，思います）。

今回用いることにしたLM1894は，その

ドルビーNR

ドルビーNRシステムというのは，ドルビー研究所というところが考え出したコンプリメンタル型のノイズリダクションシステムです。ミュージックカセットや映画の宣伝などで図1のようなマークを見たことがあるでしょう。これは，このテープ(あるいは，ここで使用するテープ)は，録音時にドルビーシステムで変調をかけていますよ，というマークです。

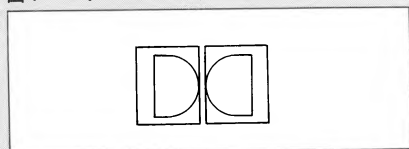
しかし，CDなどにこのマークがついているのは，あまりお目にかかりません。これは，磁気テープが「ざー」というノイズを回避できないのに対し，CDではそれができるので，下手に変調をかけるよりも原音を忠実に録音しておいたほうがよいからです。

アナログ式の磁気テープで無信号状態というのは簡単に作り出せません。必ず「ざー」というノイズが乗ってしまいます。つまり，ノイズの発生源はテープだとわかっていますから，テープに録音する前に信号に変調をかけてやるのです。具体的には，図2のとおりで，低い音はそのままに，高い音はやや大きめに，という感じで音量をいじってやります。テープのヒスノイズは周波数の高い部分に集中しており，録音のレベルにかかわらず一定の音量分しかないという特性を利用してノイズを抑えてやるのです。

このままの状態でドルビーのスイッチを切って再生してしまうと，音になることはなるのですが，高音の強調された感じの音になってしまいます。

そして，再生時にドルビーのスイッチを入れてやると，低い音はそのままで，やや高めの音は小さく変換しますので，音の高低が正確に再現されます。これが，ドルビーNRシステムの概要です。

図1　：ドルビーシステム採用のマーク

図2　ダイナミックレンジの圧縮/伸張

なかでもダイナミックノイズリダクション(以下,DNR)と呼ばれる方式を用いています。ちなみにこれを使っているものには図2のようなマークがついているはずです。この方式は,

「ある周波数帯以上の信号について,微弱な信号をカットする」

というものです。

唐突ですが,ここで質問。

1) 自分は三角関数の微分積分を知っている。

2) イマジナリショートというのは,オペアンプの2つの入力端子電圧が等しくなることであるが,そんなのは常識である。

3) どうしても今回の回路の仕組みが知りたい。

以上の質問,すべてに当てはまる人のみが,以下を読んでください。ひとつでも当てはまらない人は,次の章へどうぞ。

こういう記事の常として,動作原理なんか知らなくても,回路は動きます。回路が複雑すぎて原理を知らないとデバッグもできないような装置も世の中にはありますが,写真1を見てもわかるように,今回の装置はゴチャゴチャ考えるよりも,実際に目で回路をデバッグしていったほうが効率的です。

さて,本題に入りましょう。

まず,図4を見てください。これが今回の製作する回路のシステムブロック図です。この図からいろいろなことがわかるのですが,とりあえず細かいところから見ていきましょう。

まず,中心はメイン信号回路のAMP2です。これは明らかにローパスフィルタです。このことが明らかでない人は囲み記事をどうぞ。

中心がローパスフィルタであるということは,今回の回路は高域ノイズを除去するものだということがわかります。除去というのはいいすぎかもしれませんが,とにかく,少なくするものです。このままでも,ノイズリダクションとして機能しますが,ノイズではない高い音,たとえば,ソプラノの歌声だとか,金管楽器の音だとかも同様に,小さくなってしまいます。これはうまくありません。ということで,ほかにい

図2 ダイナミックノイズリダクション採用のマーク

図3 LM1894の内部等価回路

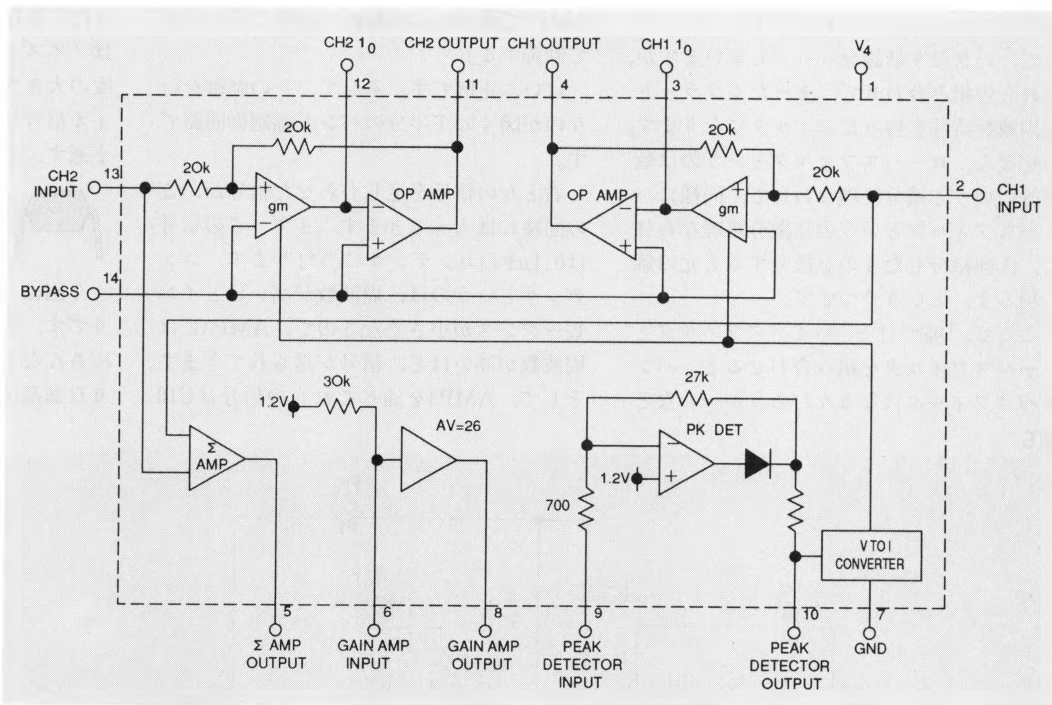
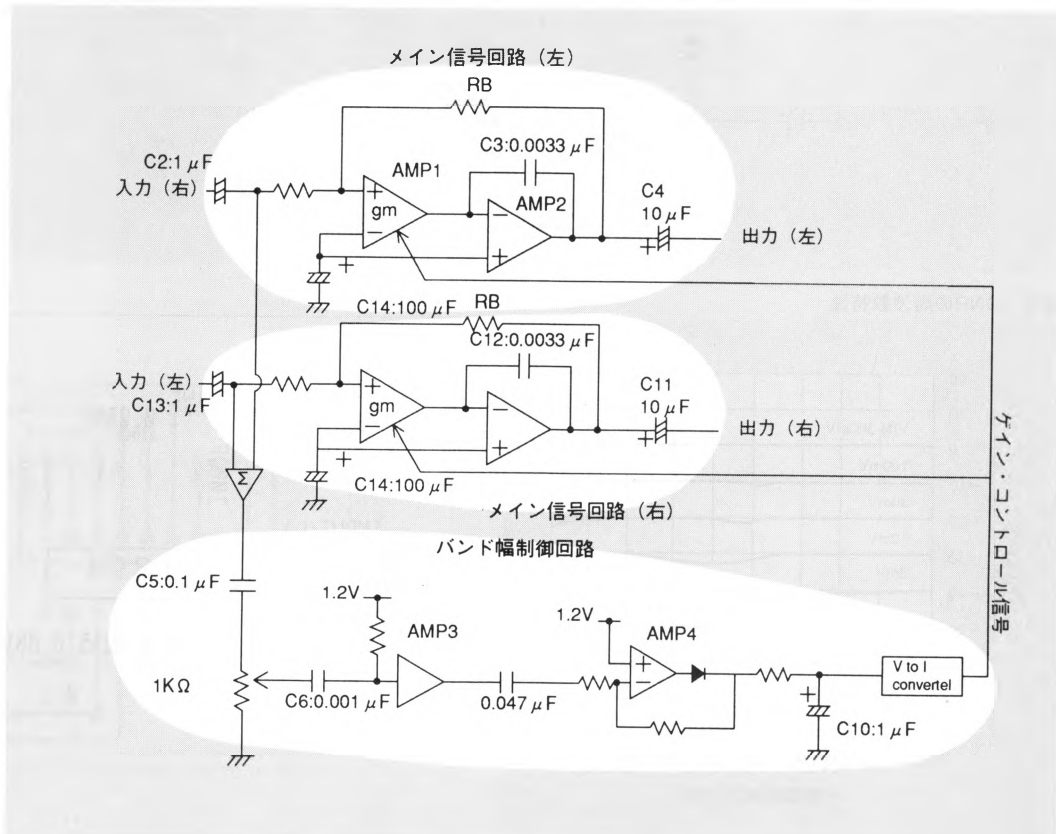
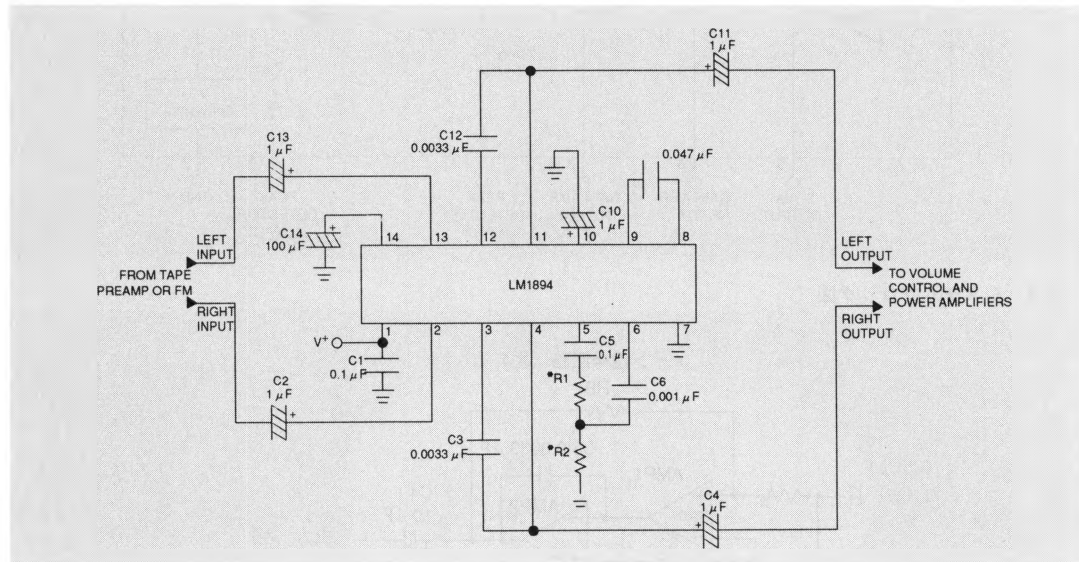
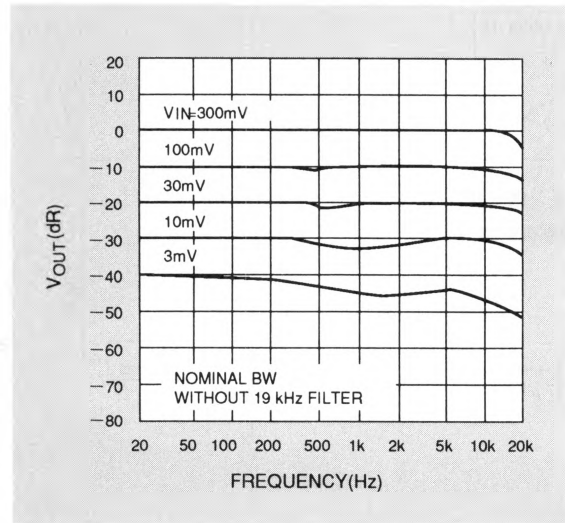
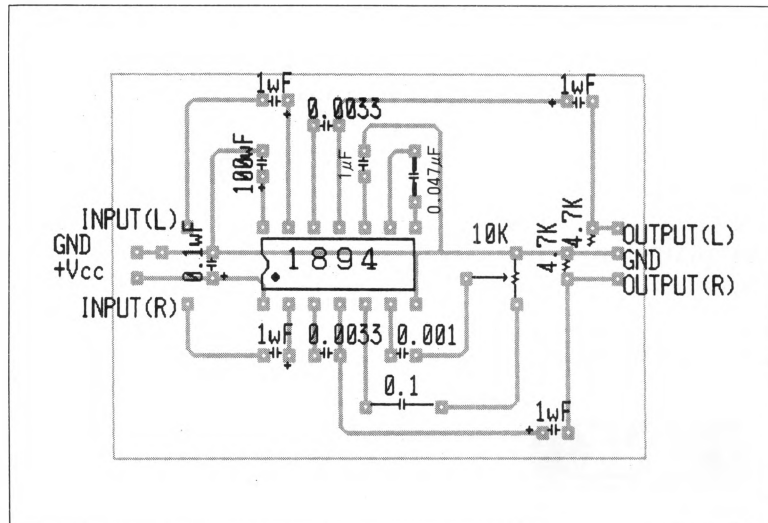



図4 システムブロック図

ろいろな回路がついています。
　このローパスフィルタの出力は，そのまま信号出力へと送られると同時に，フィードバック抵抗Rbを通じて，前段のAMP1へフィードバックされます。このAMP1をよく見ると，ハイパスフィルタに似てなくもないことがわかります。AMP2がなければハイパスフィルタそのものです。
　で，いきなり結論をいってしまいますが，これらの組み合わせで，まったくフラットな周波数特性を持ったフィルタになります。なぜなら，ローパスフィルタというのは数学的にいうと積分処理なわけで，同様にハイパスフィルタというのは微分処理なわけで，１回積分したものを微分すると元の値に戻るよ，というやつです。
　ここで，「嘘つけー，ハイパスフィルタとローパスフィルタを組み合わせると，バンドパスフィルタになるんだろうがー」などといってはいけません。あれは，カスケード接続（パソコンの用語ではデイジーチェーンというのがいちばん近いかな？）した場合の話です。今回のように入れ子構造にした場合とは別です。その証拠が図６です。
　さて，話が見えてきましたね。
「AMP2で高周波成分を除去するが，AMP1で適当にこれを打ち消すことによって制御する」
　はい，正解です。そして，その制御を行うのが図４の下半分のバンド幅制御回路です。
　右と左の信号を足し合わせた信号が，この回路には入ってきます。まず，この信号は0.1μFのコンデンサに当たります。コンデンサというのは，周波数が高いほどインピーダンスが小さくなるので，AMP3には周波数が高いほど，信号が送られてきます。そして，AMP4を通じこれらの信号はC10へと溜め込まれていきます。コンデンサに，電荷が溜まるとQ=CVですから，これは電圧信号になります。この電圧信号はV to Iコンバータを通して電流信号として，先ほどのメイン信号回路部のAMP1の制御信号となります。
　大ざっぱにいってしまうと，先ほどもいったように，このDNRは，弱い高周波成分はノイズとみなして通しませんが，ある程度の大きさに達した高周波成分は，オーディオ信号であるとみなして弱めずにそのまま通す，ということになります。

部品の入手について&製作

　今回は，連載の名前どおりICはひとつきりです。また，このLM1894以外は，パーツ屋さんならどこにでも売っているありきたりな部品ばかりです。
　部品をひと通り揃えたら，図５と電気的に同じものを作るのですが，物理的に同じ配置にできるわけはありません（コンデンサの横に抵抗があって，下にグランドがきていて……というようなこと）。
　で，一度「図５と同じものを作るんだー」と心に誓ったあと，基板の上に部品を図８のようにはめ込みます。はめ込み終わったら，抵抗やコンデンサの足を折り曲げたり，ジャンパ線（電気を通す針金のこと，かな？）を

図5

図6　DNRの周波数特性

図7

使ったりしてハンダづけします。

これで，電気的には図5と同じものができあがりました。

オーディオ信号の入力端子，出力端子に，それぞれX68000からの信号，スピーカ(あるいはイヤホン)をつなぎ，とにかく電源を入れてみます。

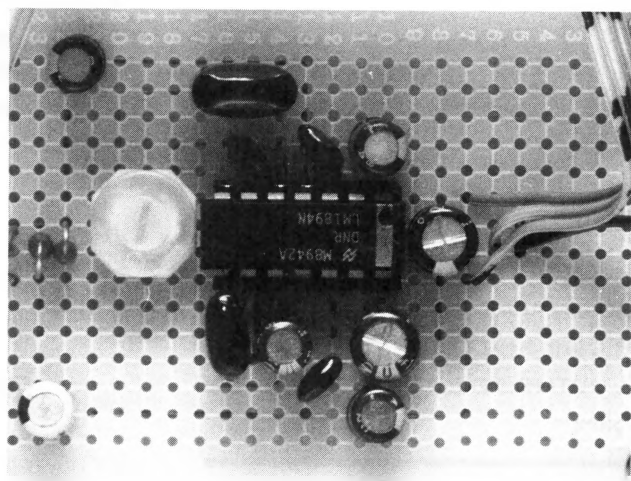
音が聞こえてくるようならば，とりあえず成功です。回路の作動スイッチを入れたり切ったりして，雑音が少なくなるのを確認してください。

回路の調整

回路の動作が確認されたなら，あとは調整を行っておしまいです。原理のところで説明しましたが（読み飛ばしたかもしれないですが，説明したのです)，今回のノイズリダクションはDNR方式といって，ある一定のレベルに満たない微弱信号はノイズだから除去しちゃえってな具合で動作します。

この「ある一定のレベル」を基板上の半固定抵抗を調節して決めてやります。これは，一応理論があって最適値があるのですが，それはノイズの発振周波数がわかっていたときの話で（ラジオとかだとどうしても音声信号のほかに，それを運んできてくれた電波のうねりのようなものが混じっちゃうでしょ。でも，それは周波数がわかっている分だけ（文化放送は1134 kHzとか）始末がよいのです)，今回は自分の耳だけが頼りです。適当に，ドライバーなどで半固定抵抗を回して，ここだっ！　と思うところで止めておいてください。きっと，そこが最適点となっていることでしょう。

次回の予告

フリーウェアで，カラオケPRO-68Kというのを見せてもらいました。結局は音楽に同期して，その歌詞が画面に表示されるというものだったのですが，その凝り方が尋常ではなく，大変感動しています（キーの調整ボタンがないのが，ちと残念)。

で，1チップにできる予定はないのですが，このプログラムに合わせて使えるようなミキサーやプリアンプを作ってみようかな，と思っています(エコーは作ったしね)。

そんなわけで，次回もお楽しみに。

参考文献

1）ナショナルセミコンダクター，LM1894データシート

2）小寺　富士夫，ノイズリダクション用ICの使い方，トランジスタ技術1987年11月号

ローパスフィルタとハイパスフィルタ

ローパスフィルタというのは，低い(ローな)周波数帯の信号をよく通す（パスする）フィルタです。じゃぁ，低くない周波数帯の信号はあまり通さないんだな，というわけでハイカットフィルタとも呼ばれています。

同じように，ハイパスフィルタ（ローカットフィルタ）は，高い周波数帯の信号をよく通し，低い周波数帯の信号はあまり通しません。

さて，図1を見てください。オペアンプの⊖入力端子は，フィードバックをかけた場合，イマジナリショートといって，⊕入力端子と等しくなっていますからこの場合0Vです。そうすると，入力側から流れ込む電流というのは，

$$I = \frac{Vi}{R}$$

です。オペアンプは入力信号の電力を消費しませんから，その電流はそのままコンデンサのほうへと流れます。コンデンサには，

$$Q = C \times V$$

Q：コンデンサの電荷
C：コンデンサの電気容量
V：電圧

という法則がありましたから，

$$Vout = \frac{Q}{C}$$

です。ここで，電流というのは電荷の変化量だと書いてあった物理の教科書を思い出して，時間あたりの変化量というのは微分値なんだという数学の教科書も思い出します。すると，

$$I = \frac{dQ}{dt}$$

ですから，

$$Q = \int I\,dt$$

となります。

そんなわけで，入力と出力の関係をまとめると，

$$Vout = \frac{1}{CR}\int Vi\ dt$$

となります。つまり，入力された電圧を積分して，1/CR倍して出力してやる回路です。1/CR倍するのはともかく，とにかく積分回路です。

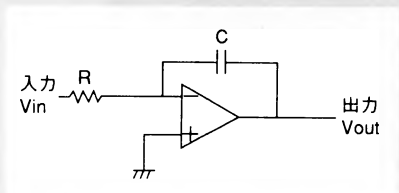
で，たとえば，

$Vi = \sin\omega t$

のような入力を入れてやれば，

$$Vout = -\frac{1}{\omega}\cos\omega t$$

のように出力されるわけです。ということは，sinとcosは位相が90度ずれているだけで，大きさに変わりはありませんから，周波数が大きければ大きいほど，ωの値が大きくなって，その逆数で出力が小さくなってしまうのです。

以上が，積分回路＝ローパスフィルタ＝ハイカットフィルタの仕組みです。

でもって，図2は微分回路＝ハイパスフィルタ＝ローカットフィルタの回路図で，積分回路の逆を行います。

$Q = C\ Vin$

で，

$$I = \frac{dQ}{dt} = C\frac{dVin}{dt}$$

となって，全然日本語になっていませんが，

$$Vout = I\ R = CR\frac{dVin}{dt}$$

です。CR倍されてはいるものの，入力電圧Vinを微分して出力する回路です。

例によって，

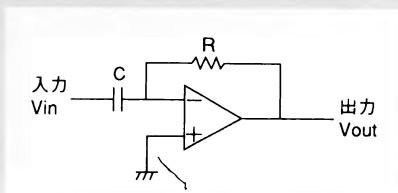
$Vin = \sin\omega t$

という入力信号は微分されると，

$Vout = \omega\ \cos\omega t$

です。つまり，周波数が大きければ大きいほど，ωの値が大きくなって，出力も大きくなるのです。ということは，相対的に周波数の低い信号は弱まってしまうことになります。

図1　ローパスフィルタ

図2　ハイパスフィルタ

ローランド SC-33

Tama Tamaki たま たまき

ローランドの新しいGS音源モジュールSC-33の試用レポートです。SC-33は低価格でSC-55よりも発音数が増やされ，エフェクタも強化されています。はたしてこれが新しいDTM音源の標準となっていくのでしょうか？

最近，楽器メーカー各社からGM対応の音源が相次いで発売されるようになりました。インテルの86系の世界ではMS-Windows 3.1でMME標準対応になり，GM対応音源が標準となりつつあります。やはり，サウンドブラスター（FM音源です）では満足できないということでしょうか？　パソコン通信にアップロードされているデータを見ても，GS/GM対応のものが多くなってきました。

ローランドのGSフォーマットはGMに対応しています。先発メーカーとしてのブランド力があるからでしょうか，SC-55はよく売れているようですし，JV-30やCMシリーズ，ローランドピアノなんかもGS対応のものが発売されています。

ということで，今回は1992年10月末に発売されたSC-55の廉価版といえるSC-33をご紹介しましょう。

SC-33とは？

まずは，挨拶代わりにスペックを。

最大同時発音数	28音(28パーシャル)
パート数	16
エフェクト	リバーブ/ディレイ，コーラス
MIDI端子	IN/OUT/THRU各1個
音声出力	L，R(標準タイプ)ヘッドホン
外形寸法	215(幅)×165(奥行)×57(高さ) mm
重量	650g

大きさはシステム手帳よりひとまわり大きいくらいですから，置くところには困らないと思います。これで単3電池6本で動けば言うことなしなんですけど，さすがにACアダプタがなければ動きません。

特徴としては，SC-55より4パーシャルほど発音数が多いこと，マルチモードのほかにシングルモードがついていること，トーンエディットできること（シングルモード時），手前についている18個のパッドでドラムやSFXを鳴らして遊べることといったところでしょうか。操作もボタンがいっぱいついているおかげでSC-55よりも扱いやすいと思います。

たとえば，トーンを選ぶときは，まず，目的のトーンのグループが書いてあるパッドを押して，右側にある▲（上向き三角)，▼（下向き三角）が書いてあるパッドで選択します。さらに，バリエーションが存在する場合は右上にあるVALUE/VARIATIONボタンで選択できます。ひとつのグループには16種類の音色が割り当てられています。

SC-33のLCDに表示される情報は現在選択されているパートの音色名とREVERB/CHORUSのON/OFFです。CM-300/500なんかは外部からMIDI情報を与えてやらないと操作できませんし，普通のシンセサイザを使っている人にも違和感はないと思いますが，DTM用の音源として見るとSC-55/155のように16パートすべての発音状況がひと目で確認できるほうがいいなぁと思いました（そんなのパソコンのディスプレイに表示されていればいいって話もありますが……)。

シングルモード

SC-33は16ティンバーの28パーシャル，DVA機能つきの音源モジュールですが，シングルモードは28パーシャルをすべてひとつのティンバーで使用する，まぁ，4～5年くらい前のシンセサイザと同じような感覚で使用するモードですね。

というわけでシングルモードは，キーボードを演奏する人用のサブ音源として使用する感が強いです（DTMではほとんど使わないと思うけど，使えないことはない)。

トーンエディットができることもシング

ローランドSC-33　39,800円

ルモードの特徴でしょう。液晶が狭いのでエディットするのは結構つらいところもあります。慣れの問題もあるでしょうけど。

シングルモードにはマルチモードにはない3つの機能があります。これらの機能はボタンひとつで切り替えることができます。

●FAT

原音に1オクターブ下の音（オクターブユニゾン）や微妙に音程のずれた音（ディチューン）などを重ねて音に厚みをつけるもので，トーンごとに設定することができます。

設定できる効果は以下の4種類です。

Octave 1：1オクターブ下の音を重ねる。

Octave 2：1オクターブ下の音と2オクターブ下の音を重ねる。

Detune 1：原音にピッチの少しずれた音を加え，ゆれの少ないコーラス効果をつける。

Detune 2：Detune 1よりもピッチのずれた音のレベルを大きくし，より広がりのあるコーラス効果をつける。

●SPLIT

指定したキー（スプリットポイント）を境に鍵域がアッパー（高音域）とロアー（低音域）に分割され，それぞれの鍵域に違うトーンを割り当てることができます。いわゆる，ひとつの鍵盤をあたかも2つの鍵盤として使用するというものです。ロアーにベース系のトーン，アッパーにブラス系のトーンを割り当てるなんてことは結構ポピュラーな使い方ですね。比較的，発音域が狭いトーンを割り当てるのが有効かと思います。

●DUAL

全鍵域にわたり，2つのトーンを割り当て，重ねて発音させるモードです。それぞれメイントーン，サブトーンと呼びます。たとえば，ストリングス系のトーンとベル系のトーンを重ねてクリスマスソング御用達のベルストリングにするということが簡単にできます。

SPLITで指定するロワートーンとDUALで指定するサブトーンには以下の制約があります。

1)　トーンレベルは各トーンで設定したトーンレベルではなく，ロアートーンレベルまたはサブトーンレベルで設定したものになります。

2)　FATは無効になります。

3)　エフェクタのうち，リバーブではタイプ/タイム/ディレイフィードバック，コーラスではディレイ/レイト/デプス/フィードバックがアッパートーンまたはメイントーンと同じ設定になります。

デジタルパーカッション

シングルモードでDRUMSボタンを押すと，SC-33はデジタルパーカッションマシンに早変わり。前面の大きな18個のパッドにはあらかじめトーンがアサインされています。もちろん各ドラムセットによってアサインが異なりますが，パッドに表記されている音に似ているものが鳴ります。アサインされているトーンはマニュアルのドラムパッド一覧表に明記されています。宴会やカラオケの盛り上げ役として，叩きまくると結構ウケるかも……？

SC-55との互換性

読者の皆さんがいちばん気にする点といえば，いままで発売されたSC-55やCM-300などとの互換性でしょう。価格がCM-300より安くなっているのに性能が上回っているのには，なにか理由があるはずです。特にスペシャルに割り当てられていたMT-32互換バンクとCM-32Pバンクが気になるところです。SC-33には残念ながらこの2つのバンクはありません。ついでにCM-64ドラムセットもありません。

純粋なGS/GM音源として見るのであれば，別に必要がないので問題ないですが，これらの音のなかには標準の音より少ないパーシャル数で発音できるものも含まれていますので，すでにこれらを多用した音楽データというのがかなり出回っています。音楽データの再生だけならデータを変更す

GS音源のひみつ

MT-32，CM-64に代わってDTMの王者に君臨したGS音源。現在発売されているだけでSC-55，SC-155，SC-33，CM-300，CM-500，JV-30，JW-50，SD-33の8種類がある。X68000対応の市販ゲームソフトのなかにも「GS音源対応」と銘打って発売されるものが多くなってきた。また巷で出回っている音楽データ，雑誌掲載される音楽データもGS音源を対象としたものが多くなってきている。しかし，困ったことに同じGS音源も微妙な違いがある。ここで現在発売中のGS音源をSC-55を基準にグループ分けしてみよう。

●SC-55対応の曲を完璧に鳴らせる

→SC-155，CM-300，CM-500。

SC-55対応の演奏データ，ゲームソフトに対して100％対応できると考えてよいだろう。実際SC-55の廉価版と位置づけられるCM-300はいまでもかなり売れていると聞く。このタイプを俗に「SC-55系音源」と呼ぶ。

●SC-55の曲をだいたい鳴らせる

→JV-30

JV-30は丁度SC-55に鍵盤がついたモデルと考えてよいだろう。SC-55対応の演奏データはほぼ間違いなく演奏できるが，大量のコントロールチェンジの送信をごくまれに取りこぼすことがあるという報告がある。バリエーション切り替えをともなった音色変更にごくまれにつまずくという話も聞く。

●SC-55の曲は鳴らせるはずだが……

→JW-50

音源スペック的には完璧な互換性があるにもかかわらずMIDIの受信処理に問題があり，市販アプリケーション，演奏データは全滅状態。詳しくは108ページ参照。

●SC-55とは違うGS音源？

→SC-33，SD-33

SC-55のバリエーションバンク127を削除し，その他のバンクに拡張音色を追加したモデル。バリエーション127を使用していないSC-55の演奏データは演奏できるが使用している演奏データは当然再現できない。

私の実感では「現在DTM界はGS音源が主流」という見方は間違っている(と思う)。実際は「SC-55系が主流」と見るべきではないだろうか。SC-55系音源はかなり普及してしまった現在，SC-33/SD-33などの新派の立場はかなり苦しいと思う。　（善）

ることで対処できますが，既存のSC-55対応のゲームでこれらの部分が結構使われており，こういったものは正常に演奏されません。これからDTMを始めようと思っている人から見ると残念なことです。しかし，時代は確実にGMに向かっていることと低価格化を考えると犠牲になるものもあるということでしょう。

逆に図1で示したとおりSC-33ではGS

表1 トーン一覧表

BANK	#	PC#	CC0#	トーン名	V
BANK1 : Piano	1	1	0	Piano 1	1
			8	Piano 1w	2
			16	Piano 1d	1
	2	2	0	Piano 2	1
			8	Piano 2w	2
	3	3	0	Piano 3	1
			8	Piano 3w	2
	4	4	0	Honky-tonk	2
			8	Honky-tonk w	1
	5	5	0	E. Piano 1	1
			8	Detuned EP 1	2
			16	E. Piano 1v	2
			24	60's E. Piano	1
	6	6	0	E. Piono 2	1
			8	Detuned EP 2	2
			16	E. Piano 2v	2
	7	7	0	Harpsichord	1
			8	Coupled Hps.	2
			16	Harpsi. w	2
			24	Harpsi. o	2
	8	8	0	Clav.	1
BANK2 : Chromatic Percussion	1	9	0	Celesta	1
	2	10	0	Glockenspiel	1
	3	11	0	Music Box	1
	4	12	0	Vibraphone	1
			8	Vib. w	2
	5	13	0	Marimba	1
			8	Marimba w	2
	6	14	0	Xylophone	1
	7	15	0	Tubular-bell	1
			8	Church Bell	1
			9	Carillon	1
	8	16	0	Santur	1
BANK3 : Organ	1	17	0	Organ 1	1
			8	Detuned Or. 1	2
			16	60's Organ 1	1
			32	Organ 4	2
	2	18	0	Organ 2	1
			8	Detuned Or. 2	2
			32	Organ 5	2
	3	19	0	Organ 3	2
	4	20	0	Church Org. 1	1
			8	Church Org. 2	2
			16	Church Org. 3	2
	5	21	0	Reed Organ	1
	6	22	0	Accordion Fr	2
			8	Accordion It	2
	7	23	0	Harmonica	1
	8	24	0	Bandoneon	2

BANK	#	PC#	CC0#	トーン名	V
BANK4 : Guitar	1	25	0	Nylon-str. Gt	1
			8	Ukulele	1
			16	Nylon Gt. o	2
			32	Nylon.Gt. 2	1
	2	26	0	Steel-str. Gt	1
			8	12-str. Gt	2
			16	Mandolin	1
	3	27	0	Jazz Gt.	1
			8	Hawaiian Gt.	1
	4	28	0	Clean Gt.	1
			8	Chorus Gt.	2
	5	29	0	Muted Gt.	1
			8	Funk Gt.	1
			16	Funk Gt.2	1
	6	30	0	Overdrive Gt	1
	7	31	0	DistortionGt	1
			8	Feedback Gt.	2
	8	32	0	Gt. Harmonics	1
			8	Gt. Feedback	1
BANK5 : Bass	1	33	0	Acoustic Bs.	1
	2	34	0	Fingered Bs.	1
	3	35	0	Picked Bs.	1
	4	36	0	Fretless Bs.	1
	5	37	0	Slap Bass 1	1
	6	38	0	Slap Bass 2	1
	7	39	0	Synth Bass 1	1
			1	Synth Bass 101	1
			8	Synth Bass 3	1
	8	40	0	Synth Bass 2	2
			8	Synth Bass 4	2
			16	Rubber Bass	2
BANK6 : Strings/Orchestra	1	41	0	Violin	1
			8	Slow Violin	1
	2	42	0	Viola	1
	3	43	0	Cello	1
	4	44	0	Contrabass	1
	5	45	0	Tremolo Str	1
	6	46	0	Pizzicato Str	1
	7	47	0	Harp	1
	8	48	0	Timpani	1
BANK7 : Ensemble	1	49	0	Strings	1
			8	Orchestra	2
	2	50	0	Slow Strings	1
	3	51	0	Syn. Strings 1	1
			8	Syn. Strings 3	2
	4	52	0	Syn. Strings 2	2
	5	53	0	Choir Aahs	1
			32	Choir Aahs 2	1
	6	54	0	Voice Oohs	1
	7	55	0	Syn Vox	1
	8	56	0	Orchestra Hit	2

BANK	#	PC#	CC0#	トーン名	V
BANK8 : Brass	1	57	0	Trumpet	1
	2	58	0	Trombone	1
			1	Trombone 2	2
	3	59	0	Tuba	1
	4	60	0	Muted Trumpet	1
	5	61	0	French Horn	2
			1	French Horn 2	2
	6	62	0	Brass 1	1
			8	Brass 2	2
	7	63	0	Synth Brass 1	2
			8	Synth Brass 3	2
			16	Analog Brass 1	2
	8	64	0	Synth Brass 2	2
			8	Synth Brass 4	1
			16	Analog Brass 2	2
BANK9 : Reed	1	65	0	Soprano Sax	1
	2	66	0	Alto Sax	1
	3	67	0	Tenor Sax	1
	4	68	0	Baritone Sax	1
	5	69	0	Oboe	1
	6	70	0	English Horn	1
	7	71	0	Bassoon	1
	8	72	0	Clarinet	1
BANK10 : Pipe	1	73	0	Piccolo	1
	2	74	0	Flute	1
	3	75	0	Recorder	1
	4	76	0	Pan Flute	1
	5	77	0	Bottle Blow	2
	6	78	0	Shakuhachi	2
	7	79	0	Whistle	1
	8	80	0	Ocarina	1
BANK11 : Synth Lead	1	81	0	Square Wave	2
			1	Square	1
			8	Sine Wave	1
	2	82	0	Saw Wave	2
			1	Saw	1
			8	Doctor Solo	2
	3	83	0	Syn. Calliope	2
	4	84	0	Chiffer Lead	2
	5	85	0	Charang	2
	6	86	0	Solo Vox	2
	7	87	0	5th Saw Wave	2
	8	88	0	Bass & Lead	2
BANK12 : Synth Pad	1	89	0	Fantasia	2
	2	90	0	Warm Pad	1
	3	91	0	Polysynth	2
	4	92	0	Space Voice	1
	5	93	0	Bowed Glass	2
	6	94	0	Metal Pad	2
	7	95	0	Halo Pad	2
	8	96	0	Sweep Pad	1

BANK	#	PC#	CC0#	トーン名	V
BANK13 : Synth SFX	1	97	0	Ice Rain	2
	2	98	0	Soundtrack	2
	3	99	0	Crystal	2
			1	Syn Mallet	1
	4	100	0	Atmosphere	2
	5	101	0	Brightness	2
	6	102	0	Goblin	2
	7	103	0	Echo Drops	1
			1	Echo Bell	2
			2	Echo Pan	2
	8	104	0	Star Theme	2
BANK14 : Ethnic	1	105	0	Sitar	1
			1	Sitar 2	2
	2	106	0	Banjo	1
	3	107	0	Shamisen	1
	4	108	0	Koto	1
			8	Taisho Koto	2
	5	109	0	Kalimba	1
	6	110	0	Bag Pipe	1
	7	111	0	Fiddle	1
	8	112	0	Shanai	1
BANK15 : Percussive	1	113	0	Tinkle Bell	1
	2	114	0	Agogo	1
	3	115	0	Steel Drums	1
	4	116	0	Woodblock •	1
			8	Castanets •	1
	5	117	0	Taiko •	1
			8	Concert BD •	1
	6	118	0	Melo. Tom 1 •	1
			8	Melo. Tom 2 •	1
	7	119	0	Synth Drum •	1
			8	808 Tom •	1
			16	Elec Perc •	1
	8	120	0	Reverse Cym. •	1

BANK	#	PC#	CC0#	トーン名	V
BANK16 : SFX	1	121	0	Gt. Fret Noise •	1
			1	Gt. Cut Noise •	1
			2	String Slap •	1
	2	122	0	Breath Noise	1
			1	Fl. Key Click •	1
	3	123	0	Seashore •	1
			1	Rain •	1
			2	Thunder •	1
			3	Wind •	1
			4	Stream •	2
			5	Bubble •	2
	4	124	0	Bird •	2
			1	Dog •	1
			2	Horse-Gallop •	1
			3	Bird 2 •	1
	5	125	0	Telephone 1 •	1
			1	Telephone 2 •	1
			2	Door Creaking •	1
			3	Door •	1
			4	Scratch •	1
			5	Windchime •	2
	6	126	0	Helicopter •	1
			1	Car-Engine •	1
			2	Car-Stop •	1
			3	Car-Pass •	1
			4	Car-Crash •	2
			5	Siren •	1
			6	Train •	1
			7	Jetplane •	2
			8	Starship •	2
			9	Burst Noise •	2
	7	127	0	Applause •	2
			1	Laughing •	1
			2	Screaming •	1
			3	Punch •	1
			4	Heart Beat •	1
			5	Footsteps •	1
	8	128	0	Gun Shot •	1
			1	Machine Gun •	1
			2	Lasergun •	1
			3	Explosion •	2

\# ：ナンバー
PC# ：プログラム・ナンバー
CC0# ：コントロール・ナンバー0の値（GSバンク・セレクト・ナンバー）
V ：使用ボイス数

部分で39トーンも拡張されているということがウリでしょう。しかし，これらのトーンを使用した曲データをSC-55やCM-300で演奏させるとキャピタル（サブ含む）に変換されますので，曲の雰囲気が若干変わるおそれがあります。これらの新しい音のなかには従来のMT-32互換バンクに入っていたものもありますので，対応するものがあれば手作業で変換してやることにより，より忠実な再生をさせることもできます。

また，SC-33はSC-55より４パーシャルも発音数が多いので，SC-33では正常に演奏できる曲データでもSC-55では音切れが発生することも考えられます。

LIVE inに投稿する際にはSC-33で拡張されたトーンを使用した曲データはGS用SC-33にて作成，126，127バンクを使用した曲はSC-55/CM-300用と明記したほうがよいでしょう。

GM音源もメーカーによって，音源チップやサンプリング手法により音のニュアンスが違っていることがあります。特にエフェクタの機能がメーカーによってまちまちなのでちょっとコントロールチェンジでリバーブをいじくっただけでほかのGM音源ではボリュームのバランスが狂ってしまいます。したがって，GM音源用の曲データについても同様に機種名を明記してGM音源用としたほうがよいでしょう。

冒頭に書いたとおり，MS-DOSマシンの世界では着実にGM/GSに傾きつつあること，ローランドやヤマハなどがGM/GS用のスタンダードMIDIフォーマットの曲データを販売していることなどを考えると，これからはSC-33のスペックで十分だと思います。しかし，過去の資産（これが頭の痛いところ）と価格を考慮するとCM-300のほうが有利かなぁ，と思う点もあります。SC-55のハーフラックサイズという点も捨てがたい気がしますし。

互換性で第一に問題になるのはゲームですから，ゲーム音楽を重視する人にはSC-55かCM-300をおすすめします。そのほかの性能に関してはSC-33のほうが上です。このスペックがGSの標準となるのか，やはりSC-55相当のものが標準となるのかは今後の展開にかかっています。すでにSC-55相当の音源がかなりの出荷台数に達していることを考えると実に微妙な問題といえます。

デザインにかかわらずに，機能だけでいうならば，CM-64ユーザーでGS/GM音源を購入しようと思っている人は迷わず買いでしょう。

MT-32/CM-32LだけのユーザーだとSC-55のCM-32Pバンクがほしいところですが，CM-64用の曲データはたいがいRS-PCMカードを使用したものが多いので，その点を犠牲にしてもSC-33を買う価値はあると思います。

安いサブ音源を探しているキーボード弾きの人も迷わず買いです。理由はDTM用のGS音源とライブ用のサブ音源がSC-33ひとつで実現できるからです。

対抗馬にヤマハのTG-100という約５千円も安いGM音源がありますが，SC-55とTG-100を比べるのであればメーカー希望小売価格ベースが２万円以上も開きがあるし，TG-100も考慮すべきかな？と思いましたが，SC-33の登場でTG-100がそれほど魅力的だと感じなくなりました。店頭販売価格で比べればSC-33とTG-100の価格差なんて微々たるものになっているでしょう。また，河合楽器のGMegaも人気があるようですが（ハーフラックサイズなんだよね），エントリーモデルとして考えるとSC-33のほうがよいと思います。

最後にローランドに期待することといえば，SC-33のシングルモードなしの低価格音源を３万円台で出してほしいということだけです。1993年のGM/GS音源のエントリーモデルは４万円台の攻防になることは，まず間違いないでしょう。いやぁ，いい時代になったものです。

表2

PC#	CC0	SC-33で拡張されたバリエーション	SC-55/CM-300で変換されるキャピタル
1	8	Piano 1w	Piano 1
1	16	Piano 1d	Piano 1
2	8	Piano 2w	Piano 2
3	8	Piano 3w	Piano 3
4	8	Honky-tonk w	Honky-tonk
5	16	E.Piano 1v	E.Piano 1
5	24	60's E.Piano	E.Piano 1
6	16	E.Piano 2v	E.Piano 2
7	16	Harpsi.w	Harpsichord
7	24	Harpsi.o	Harpsichord
12	8	Vib.w	Vibraphone
13	8	Marimba w	Marimba
15	9	Carillon	Church Bell
17	16	60's Organ 1	Organ 1
17	32	Organ 4	Organ 1
18	32	Organ 5	Organ 2
20	16	Church Org.3	Church Org.1
25	16	Nylon.Gt.o	Nylon-str.Gt
25	32	Nylon.Gt.2	Nylon-str.Gt
29	16	Funk Gt.2	Muted Gt.
39	1	SynthBass101	Synth Bass 1
40	16	Rubber Bass	Synth Bass 2
41	8	Slow Violin	Violin
53	32	Choir Aahs 2	Choir Aahs
58	1	Trombone 2	Trombone
61	1	French Horn 2	French Horn
63	16	AnalogBrass1	Synth Brass1
64	16	AnalogBrass2	Synth Brass2
81	1	Square	Square Wave
82	1	Saw	Saw Wave
82	8	Doctor Solo	Saw Wave
99	1	Syn Mallet	Crystal
103	1	Echo Bell	Echo Drops
103	2	Echo Pan	Echo Drops
105	1	Sitar 2	Sitar
119	16	ElecPerc	Synth Drum
124	3	Bird 2	---------------------------

SC-33	SC-55/CM-300 etc
33 JAZZ Set,41 BRUSH Set 35 JAZZ KD 2	Kick Drum 2
33 JAZZ Set,41 BRUSH Set 36 JAZZ KD 1	Kick Drum 1

新製品紹介　GS音源搭載のミュージックワークステーション

ローランドJW-50

Nishikawa Zenji 西川 善司

SC-55系GS音源に鍵盤をつなぎ，さらにシーケンサやFDDまで組み込んだミュージックワークステーション，それがJW-50です。DTMという分野には必ずしも適しているとはいえませんが，GS音源用データの作成には大いに力を発揮しそうです。

JW-50とは？

ローランドJW-50は，人気のGS音源モジュールSC-55系(SC-155/CM-300)に16トラックシーケンサと鍵盤を合体させたモデルと思ってもらえばよいだろうか。

同時発声数は24。受信MIDIチャンネルは16，つまり16マルチティンバー。プリセット音色はまったくSC-55と同じ。もちろん，リズムキットの配列や種類も同じである。カタログには詳しく言及はされていないが，SC-33では削除されてしまったMT-32の互換音色セットであるバリエーション127もちゃんとある。

エクスクルーシブメッセージに関しても99%の互換性がある。受信MIDIチャンネルやパーシャルリザーブなどのパートパラメータはもちろん，音色に関するパラメータのアドレスの配列もまったく同じである。1%の違いとは，SC-55にあった画面表示関係のアドレスがないことくらいである。

では，JW-50を構成している各セクションをひとつずつ見ていくことにしよう。

音源部

SC-55相当の音源がまるごと音源部に入っていると考えて支障はないようだ。しかし，バンク10369をユーザー音色セットに割り当てられるという点が拡張されている。従来SC-55系音源は音色をエディットすることが可能であったが，ちょっとしたことで設定したパラメータがリセットされてしまい設定が失われてしまうという悲惨な設計であった。JW-50では，ユーザーがエディットした音色は128個まで記憶させることができ，これをプリセット音と同等に使って曲を作成することができる。SC-55系にもぜひほしかったうらやましい機能である(シンセサイザならば本来ついていて然るべき機能なんだが)。

シーケンサ

シーケンサは16トラックと一般的なものが入っている。リアルタイムレコーディングはもちろんだが，マルチチャンネルリアルタイムレコーディングの機能も搭載されている。これは，複数のMIDIチャンネルからの入力をレコーディングしてしまうというもので，たとえば，複数人による演奏もレコーディングできるということだ。また，別のシーケンサ(X68000でもいいが)の演奏を取り込むという使い方もできる。すなわち既存の演奏データをJW-50に落とすといった使い方も可能なわけだ。もちろん，クォンタイズや移調，パラメータシフト，カットなどのレコーディング後の編集機能も充実している。

もちろん，ステップ入力機能もある。♪=60クロックというこのランクでは標準的な解像度だ。複写，削除機能も基本的な機能は揃っており，うれしいのはピッチベンダーやコントロールチェンジ，エクスクルーシブまでもが，各ノートのあいだにイベントとして挿入することができる点だ。これにより，かなり凝った曲も作成できそうだ。

ミキサー

シーケンサには本格的なコンピュミキサ機能もついている。シーケンサで作成した曲に対して，本体についている8本のフェーダを動かすことによって，パンポット，音量などのパラメータをリアルタイムに変化させることができ，さらに，これを曲データに効果として記憶させることができるのだ。たとえば，曲の最後をフェードアウトしたいとき，演奏終了直前にこのフェーダで音量を下げてやるだけでよい。すると，次回の演奏からは，そのタイミングでこのフェーダの動きが再現されるのである。

バッキング

JW-50には手軽にオリジナルソングが作れるようにと，自動バッキング機能というユニークな機能が備わっている。これは，さまざまな音楽ジャンルの基本的なパターンの伴奏を行ってくれるというものだ。ユーザーはこれにあわせてメロディなどの味つ

ローランドJW-50

けをすれば即興でオリジナルソングが作れるというわけだ。バッキングパターンはロック，ポップス，ジャズ，フュージョンからバラード，ワルツ，ボサノバ，サンバまで多種多様にわたっており，全部で30種類から選べる。もちろんユーザーが作成したバッキングパターンを登録しておくこともできる。

実際にバッキング機能の演奏を聞いてみたが，なかなかいい演奏が揃っている。というのも，ベース，リズム，インストパート２パートの全部で４パートで構成された演奏で，それ単体である程度完成されているからだ。逆にいえば，このバッキング機能を使って作った曲はみな似たりよったりになってしまいそうということだが，かなり遊べる機能ということで高く評価はしたい。

フロッピーディスクドライブ

フロッピーディスクドライブ(以下FDD)は2DD(容量約640Kバイト)を採用している。このFDDではJW-50のさまざまな設定やデータ（曲データ，ユーザー音色，ユーザーバッキングパターンなど）を保存することができる。

また，スタンダードMIDIファイル(以下SMF)を扱う機能がついているのがうれしい。市販のSMF演奏データ集をロードして演奏できることはもちろん，シーケンサ画面でエディットしたりすることもできる。また，JW-50で作成したオリジナルソングをSMFへコンバートすることもできる。MUSIC WORKSTATIONの名前にふさわしい機能だ。

JW-50の問題点

これはいっておかなくてはならないだろう。JW-50はMIDIの受信速度が極端に遅い。おそらくSC-55と同等のプロセッサでシーケンサやミキサの処理まで行っているので処理速度が追いつかないのだろう。

SC-55と音源的にみれば完璧な上位互換を持っていながら，MIDI受信に関しては残念ながら完全な下位互換である。ちょっとMIDIデータを多めに送信すると処理しきれずにすぐ「MIDI error」を発生してしまうのだ。パーシャルリザーブなどの基本的なエクスクルーシブメッセージを送信してもすぐにこのエラーを発生してしまい，もはや，JW-50をSC-55系の音源の代わりとして使うのは無理と考えてもよいだろう。

SC-55系用の演奏データというのはエクスクルーシブメッセージやコントロールチェンジを駆使した大変凝った作りのものが多い。「SC-55対応」の市販ゲームソフトなどのBGMはなおさらそういった傾向が強い。こういった演奏データに対応できないJW-50は最近のDTMの常識を大きく逸脱したマシンといわれてもしかたはないだろう。実際に，

コナミ「出たな!! ツインビー」
コナミ「グラディウスII」
ズーム「オーバーテイク」

といった「SC-55対応」の市販ゲームソフトを試してみたが正常な演奏ができなかった。本体の機能が大変充実しているだけに実に残念な欠点だ。ぜひ改善してもらいたい。

結論

１台でなんでもかんでもできるマシンということで１台目のシンセサイザとしてはオススメできる。しかし，DTMという用途に使用することはあまり考慮されていない。SC-55の代わりに……というのは前段の欠点からもわかるように奨励できない。ただし，JW-50の演奏データはSC-55系で完全に再生できるということから，GS音源用曲データの制作ツールとして，また，GSデータの編集ツールとしてのJW-50の価値は高いといえよう。
どちらかというとすでにSC-55を持っていてデータ作成に使用したいという人にむいている楽器である。

表1　バッキングパターンの例

No.	ミュージック・スタイル名／プリセット・コード・チェンジ名	コード（オリジナル／バリエーション1／バリエーション2）							
1	Rock 1	Em	>Em	>GMaj	>GMaj	>AMaj	>AMaj	>BMaj	>BMaj
		Fm	>Fm/A♭	>E♭Maj/C	>Cm7	>Fm	>Fm7/A♭	>E♭7	>E♭7
		Dm	>FMaj	>GMaj	>Dm	>Dm	>FMaj	>GMaj	>Dm
2	Rock 2	CMaj	>CMaj	>E♭Maj	>E♭Maj	>F7	>F7	>G7	>G7
		Am	>FMaj	>GMaj	>Am	>Am	>FMaj	>GMaj	>Am
		Cm7	>Cm7	>E♭Maj	>FMaj	>Cm7	>Cm7	>Cm7/G	>Cm7/G
3	Rock 3	E7	>E7	>E7	>E7	>AMaj	>AMaj	>B7	>B7
		CMaj	>CMaj	>C7	>C7	>E♭Maj	>E♭Maj	>FMaj	>FMaj
		Em7	>Em7	>Em7	>Em7	>Em7	>Em7	>Em7	>G7
4	Triplet Rock	E♭Maj	>FMaj	>FMaj/C	>CMaj	>E♭Maj	>FMaj	>FMaj/C	>CMaj
		CMaj	>CMaj	>B♭Maj	>B♭Maj	>E♭Maj	>E♭Maj	>A♭Maj	>G7
		CMaj	>CMaj	>FMaj	>FMaj	>CMaj	>CMaj	>G7	>G7
5	Funk	C7	>C7	>C7	>C7	>C7	>C7	>C7	>C7
		Cm7	>Cm7	>Cm7	>Cm7	>Cm7	>Cm7	>Cm7	>Cm7
		Dm	>Dm	>Dm	>Dm	>C7	>C7	>C7	>C7
6	Brass Funk	E7	>E7	>D7	>D7	>GMaj	>GMaj	>A7	>A7
		E7	>E7	>E7	>E7	>E7	>E7	>E7	>E7
		CMaj/D	>DMaj	>FMaj/D	>DMaj	>CMaj/D	>DMaj	>FMaj/D	>DMaj
7	R & B	D7	>D7	>C7	>C7	>D7	>D7	>D7	>D7
		D7	>D7	>C7	>C7	>Em	>Em	>D7	>D7
		C7	>C7	>Dm	>G7	>C7	>C7	>C7	>C7
8	Fusion 1	C69	>C7/E	>FM7	>Fm7/B♭	>Am9	>D9	>Dm7/G	>G7
		Em9	>Em9	>F#m11	>B7	>Em9	>Em9	>F#m11	>B7
		CM7	>CM7	>Bm7	>Em7	>Am7	>D7	>GM7	>C#7
9	Fusion 2	Bm11	>Bm11	>Bm11	>Bm11	>B♭M7	>B♭M7	>B♭M7	>B♭M7
		Fm7	>B♭7	>E♭M7	>A♭M7	>Am7♭5	>D7	>Gm7	>F#m7
		Gm7/C	>Gm7/C	>Gm7/C	>Gm7/C	>B♭m7/E♭	>B♭m7/E♭	>B♭m7/E♭	>B♭m7/E♭
10	Jazz Funk	Cm7	>Cm7	>Cm7	>Cm7	>Cm7	>Cm7	>C7	>C7
		F7	>F7	>F7	>F7	>F7	>F7	>F7	>F7
		G7	>G7	>G7	>G7	>G7	>G7	>G7	>G7

X68000・Z-MUSIC用 ©Nintendo F-ZEROより **MUTE CITY** Shindo Noriyuki 進藤 慶到

X68000・Z-MUSIC用 〔SC-55対応〕 ©CAPCOM ストリートファイターIIより **ケンのテーマ** Nakazato Kazunori 中里 和紀

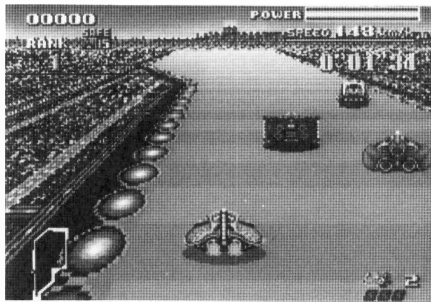
X1・MusicBASIC用 **晴れたらいいね** Abe Toshimitu 阿部 俊光

どうも最近ご無沙汰ぎみだったX1ですが，人気絶好調のドリカムで堂々の登場です。ユーザーの皆さん，お待ちどうさまでした。X68000用は人気ゲームから2曲。ストIIはなんとこれで3曲目。めざせ！　全曲。ということで，LIVE in '93も絶好調です。

ああ，夢の1分58秒台……

今日も夜が明ける。私はスーパーファミコンのコントローラを置き，軽くため息をついた。「あそこのカーブをうまくクリアしていたら……くう」。

何だか，わけのわからない書き出しですが，F-ZEROで遊んだことのある人ならお心当たりがあるでしょう。え，F-ZEROをやったことないって？　やだなあ。F-ZEROは，あの超有名ゲームの移植版や大人気RPGの続編よりも，さらに面白いかもしれないのですよ(好みもありますけどね)。

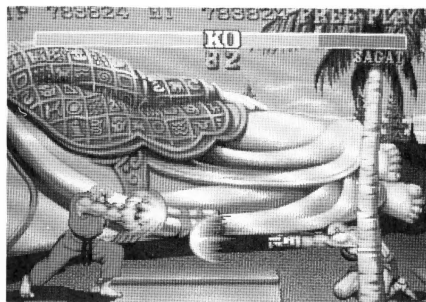
というわけで私は大ファンなんですよね，F-ZEROの。で，今回はそのなかで使われているBGMから1曲選んで作ってみました。MUTE CITY系のステージで使われている曲です。比較的短い(当社比)リストですから，F-ZEROファンの人も，そうでない人も，ぜひ聴いてくださいね。特に，MUTE CITYを死ぬほど走った人が，懐かしい気持ちになってくれればうれしいです。ひょっとすると悪夢の日々が蘇るかな(苦情は受け付けかねます……)。

演奏には，いつものとおりX68000とZ-MUSICが必要です。今回は内蔵音源だけで聴けますからご安心を。それから，PCM8.Xは組み込んでも組み込まなくても，どちらでも結構です。あったほうが，リリースが重なって自然なのですが，たいした差はありません。

さて，曲のほうはどうかといいますと，いつもどおりのパターン。原曲の持つ音のイメージなんかを大切にしてみました。逆に，相手の音源はPCMでエフェクタ搭載と手ごわいので，ところどころ苦労することもありましたけどね(家庭用ゲーム機もいい音を出すようになったものです)。

それにしても，MUTE CITYってシンプルなのになんて奥が深いのでしょう。誰か私に，あの隙間をくぐり抜ける方法を伝授してくれませんかね。　(進藤慶到)

F-ZERO

ブラック魔王？

そりゃケンケンや。失礼しました。X68000用の2曲目はストリートファイターIIより「ケンのテーマ」です。正式名称では「アメリカ(ケン)」というようですが，ケンのテーマというほうがとおりがいいですね。演奏にはZ-MUSICシステムとSC-55が必要になります。

このLIVEのページでも，ストIIからはリュウ(1992年2月号)，バルログ(1992年11月号)のテーマに次いで3曲目ということになりますね。このまま続くと，音楽が全部揃う日も遠くはないかもしれませんね。常連さんのなかにはエンディング好き(？)の西本くんとかもいますからね(笑)。カプコンさんからのリリースとどっちが早いんでしょうかね。いつまでも待ってますよ，カプコンさん。

ストリートファイターII

さて，曲のお話にいきましょう。このケンのテーマは数あるストIIミュージックのなかでも人気度はピカ一の部類に入ります。投稿されてくる曲が多いことでもよくわかります。そのなかでも，群を抜いて素晴らしかったのがこの作品です。原曲のイメージをさらに広げるようなアレンジが施されていて，逆に原曲が物足りなくなるくらいなのです。ストIIのCDにある「春雷」というアレンジバージョンにおけるケンの部分のイメージで全体をまとめあげているようですね。SC-55ユーザーはもちろん，CM-300/CM-500ユーザーもぜひとも入力してみてください。

実は1992年2月号に掲載した「リュウのテーマ」もこの中里くんの作品です。入力して聴いた人も多かったですよね。今回も安心して聴いてくださいね。

山へ行こう

ひっさしぶりにX1用の作品の登場です。しかも曲はDreams Come Trueの最新アルバム「The Swinging Star」からの選曲で,「晴れたらいいね」です。某国営放送の連ドラ「ひらり」のオープニングテーマということでも話題になりました。ってことで,ドリカムを知らない人はまずいないでしょう。実力派としてメキメキ頭角を現してきています。アルバムも5枚目になり,曲風もドリカム流として落ち着いてきたようですね。もしカラオケで吉田美和のヴォーカルを完璧にこなす人がいたら拍手をしましょう。基本的には難しい歌ばかりですから。ちなみにドリカムステップはどの曲でもほぼ一定ですので,一度覚えると応用が利いて便利ですよ。恥を捨てて大げさにやるとウケがいいかもしれません。

演奏にはMusicBASICが必要になります。それからPSGも使用しているので,ミキシングには注意してくださいね。

作品の注目点といえば,やはり上記のとおりにヴォーカルでしょう。どこまで表現できるかが腕の見せどころになります。この作品の場合,ボワーっとしたリードはサビに合わせて作られているような感じですが,違和感もなく,うまくまとめられています。う～ん,Feel so good!。

作者の阿部くんは今回で4回目の登場になります。曲はリンドバーグ→T-SQUARE→プリプリ→ドリカムという流れになっています。うむうむ,このメンツは1つのパターンですね。似たようなパターンとしては,サザン→ユーミン→ZOO→ドリカムとか,コメコメ→爆風→ドリカムなんてパターンもありますね。どのパターンにもドリカムが入ってしまうというのはすごいことかもしれません。まさに定番ということでしょうか。ほかにもよくあるパターンというのがあったら教えてくださいね。(S.K.)

The Swinging Star

またまたごめんなさい

1月号で紹介したセーラームーンの「ムーンライト伝説」ですが,入力してみて「あれれ」と思った方,本当にごめんなさい。カウンタ表示が掲載されていませんでしたね。今月号の115ページに掲載したリスト7を使ってください。セーラームーンファンの方もそうでない方も「月に代わってお仕置き」なんていわないで,許してね。

リスト1 MUTE CITY

```
  1: .comment -F·ZERO- MUTE CITY (C)Nintendo  Programed by ENG 93/01/19
  2:
  3: / for ZMUSIC.X
  4:
  5: /-------------------------------------------------------------
  6: / TRACK SETUP
  7:
  8: (i)
  9:
 10: / OPM & ADPCM
 11:
 12: (m01,1000)(aFm1,1)
 13: (m02,1000)(aFm2,2)
 14: (m03,1000)(aFm3,3)
 15: (m04,1000)(aFm4,4)
 16: (m05,1000)(aFm5,5)
 17: (m06,1000)(aFm6,6)
 18: (m07,1000)(aFm7,7)
 19: (m08,1000)(aFm8,8)
 20: (m09,1000)(aAdpcm,09)
 21: (m10,1000)(aAdpcm,10)
 22:
 23: /-------------------------------------------------------------
 24: / ADPCM DATA SET
 25:
 26: .adpcm_block_data = MUTE_CITY
 27:
 28: /-------------------------------------------------------------
 29: / OPM DATA SET
 30:
 31: /       AR 1DR 2DR  RR 1DL  TL  RS MUL DT1 DT2 AME  BASS 1
 32: (@1,    15,  0,  0,  0,  0, 28,  0,  4,  7,  0,  0
 33:         14,  0,  0,  6,  0,  3,  0,. 4,  7,  0,  0
 34:         16,  0,  0,  0,  0, 36,  0,  4,  3,  0,  0
 35:         15,  0,  0,  6,  0,  3,  0,  4,  3,  0,  0
 36: /       AL  FB  OM PAN
 37:          4,  3)
 38:
 39: /       AR 1DR 2DR  RR 1DL  TL  RS MUL DT1 DT2 AME  BASS 2
 40: (@2,    31, 15,  1,  0,  1, 39,  0, 10,  0,  0,  0
 41:         31,  4,  1,  0,  3, 30,  0,  1,  0,  0,  0
 42:         31,  4,  1,  0,  1, 33,  0,  0,  0,  0,  0
 43:         22,  7,  6,  6,  1,  3,  1,  1,  0,  0,  0
 44: /       AL  FB  SM PAN
 45:          0,  7, 15)
 46:
 47: /       AR 1DR 2DR  RR 1DL  TL  RS MUL DT1 DT2 AME  GRO
 48: (@3,    31, 14,  0,  8,  7, 40,  0, 15,  3,  0,  0
 49:         31,  7,  0,  7, 15, 12,  0,  1,  7,  0,  0
 50:         31,  0,  0,  7,  0, 36,  0,  1,  7,  0,  0
 51:         31,  7,  0,  7, 15,  8,  0,  1,  3,  0,  0
 52: /       AL  FB  SM PAN
 53:          4,  5, 15)
 54:
 55: /       AR 1DR 2DR  RR 1DL  TL  RS MUL DT1 DT2 AME  BRASS 1
 56: (@4,    20,  3,  1,  0,  1, 30,  0,  1,  7,  0,  0
 57:         25,  1,  0,  7,  1,  5,  1,  1,  7,  0,  0
 58:         20,  2,  1,  4,  1, 23,  0,  1,  3,  0,  0
 59:         26,  1,  0,  7,  1,  5,  1,  2,  3,  0,  0
 60: /       AL  FB  SM PAN
 61:          4,  7, 15)
 62:
 63: /       AR 1DR 2DR  RR 1DL  TL  RS MUL DT1 DT2 AME  BRASS 2
 64: (@5,    20,  0,  0,  0,  0, 29,  0,  1,  0,  0,  0
 65:         20,  0,  0,  0,  0, 32,  0,  1,  0,  0,  0
 66:         20,  0,  0,  0,  0, 63,  0,  8,  0,  0,  0
 67:         18,  0,  0,  8,  0,  4,  0,  1,  0,  0,  0
 68: /       AL  FB  SM PAN
 69:          2,  7, 15)
 70:
 71: /       AR 1DR 2DR  RR 1DL  TL  RS MUL DT1 DT2 AME  BRASS 3
 72: (@6,    12,  0,  0,  5,  1, 32,  3,  1,  7,  0,  0
 73:         12,  0,  0,  6,  0,  8,  3,  2,  7,  0,  0
 74:         13,  6,  0,  5,  1, 23,  2,  1,  3,  0,  0
 75:         13,  0,  0,  6,  0, 13,  2,  1,  3,  0,  0
 76: /       AL  FB  SM PAN
 77:          4,  7, 15)
 78:
 79: /       AR 1DR 2DR  RR 1DL  TL  RS MUL DT1 DT2 AME  BRASS 4
 80: (@7,    25, 11,  0,  3,  1, 33,  0,  4,  7,  0,  0
 81:         25,  0,  0,  8,  0,  9,  0,  4,  3,  0,  0
 82:         25,  0,  0,  3,  0, 32,  0,  4,  3,  0,  0
 83:         25,  0,  0,  7,  0,  9,  0,  4,  7,  0,  0
 84: /       AL  FB  SM PAN
 85:          4,  6, 15)
 86:
 87: /       AR 1DR 2DR  RR 1DL  TL  RS MUL DT1 DT2 AME  TOM
 88: (@8,    31,  0,  0,  0,  0, 13,  0,  1,  0,  0,  0
 89:         26, 23, 16,  6,  9,  5,  0,  1,  0,  2,  0
 90:         31, 17, 17,  8,  3,  7,  0,  2,  3,  0,  0
 91:         31, 16, 16,  8,  2,  3,  0,  1,  7,  0,  0
 92: /       AL  FB  SM PAN
 93:          6,  7, 15)
 94:
 95: /-------------------------------------------------------------
 96: / MML DATA SET
 97:
 98: (o196)
 99:
100: (t1)    r*10
101: (t2)    r*11
102: (t3)    r*10 @s6,6 @h26,20
103: (t4)    r*10 @s6   @h26
104: (t5)    r*11
105: (t6)    r*10 @s4,6 @h22,12
106: (t7)    r*10 @s4,6 @h22,12
107: (t8)    r*12
108: (t9)    r*10
109: (t10)   r*10
110:
111: /--------------------------------
112: / BASS
113:
114: / (t1)と(t2)は，ほとんど同じです。
115:
116: (t1)     L8@2o3@v122p3@k00@q8
117: (t1)     |:c1f1|e-1f^2d4d-:|>b-1<c^2cc>b-<
118: (t1)     |do|L8@2o3@v122p3@k00@q8
119: (t1)     |:c1f1|e-1f^2d4d-:|>b-1<c^2cc>b-<@q16^2
120: (t1)     |:3|:cccccd>b-g4gggggg<:||
121: (t1)     |:c+c+c+c+c+d+c+>g+4g+g+g+g+g+g+g+<:|:|
122: (t1)     >bbbbb<c+>bf+1|:3bbbb<c+>b|f+4:|f+
123: (t1)     e-e-e-e-ffffbbbbbbbb
124: (t1)     <f+f+f+f+c+c+c+c+>bbbbbbbbbbbb<c+c+c+c+
125:
126: (t2)     L8@2o3@v122p1@k07@q8
127: (t2)     |:c1f1|e-1f^2d4d-:|>b-1<c^2cc>b-<
```

```
128: (t2)     [do]L8@2o3@v122p1@k09@q8
129: (t2)     |:c1f1|e-1f^2d4d-:|>b-1<c^2cc>b-<@q16~1
130: (t2)     |:3|:cccccd>b-g4gggggggg<:||
131: (t2)     |:c+c+c+c+c+d+c+)g+4g+g+g+g+g+g+g+<:|:|
132: (t2)     >bbbbb<c+>bf+4|:3bbbb<c+>b|f+4:|f+
133: (t2)     e-e-e-e-ffffbbbbbbbb
134: (t2)     <f+f+f+f+c+c+c+c+>bbbbbbbbbbbb<c+c+c+c+
135:
136: /-------------------------------
137: / MAIN
138:
139: / (t3)と(t4)は，前半と最終行が違うだけです。
140:
141: (t3)     L16@3o4@v109p3@k-2q7
142: (t3)     |:7g_3g~3:||:8a_3a~3:||:8b-_3b-~3:||:9a_3a~3:|
143: (t3)     |:7b-_3b-~3:||:8a_3a~3:||:8b-_3b-~3:||:9a_3a~3:|
144: (t3)     [do]L16@3o4@v109p3@k-2q6@m@a
145: (t3)     |:7g_3g~3:||:8a_3a~3:||:8b-_3b-~3:||:9a_3a~3:|
146: (t3)     |:7b-_3b-~3:||:8a_3a~3:||:8b-_3b-~3:||:9a_3a~3:|
147: (t3)     L4@6o5@v127p3@k00@q6@m9@a3
148: (t3)     |:
149: (t3)     @q8f.e-.@q4d@q8c.d.@q4e-
150: (t3)     @q8f.e-.@q4d@q8c2r@q6{cde-f}
151: (t3)     @q8g.f.@q4_1b-@q8a-.~1g.@q4f
152: (t3)     @q8g.f.@q4e-@q8d.c.>@q4b-<
153: (t3)     :|@q8
154: (t3)     L8>a4r4.gab-<c4r4.>gab-<d4r4.>ab-<de-4r4.>b-<de-
155: (t3)     f4r4.>a-<d-e-f4r4.d-fa-<d-4r4.>fa-<d-f4r4.>a-<d-f
156: (t3)     @q6g-.f.e-d-.>b.<d->b-.a-.g-f.e-.d-e-.f.g-e-.f.b-
157: (t3)     <e-1r2_1{g-edd->ba}2
158:
159: (t4)     L8@1o4@v118p3@k00@q7
160: (t4)     |:|:15c:||:8e-:||:9c:|:|
161: (t4)     [do]L8@4o4@v118p3@k00@q7@m
162: (t4)     |:|:15c:||:8e-:||:9c:|:|r*14
163: (t4)     L4@6o5@v120p3@k05@q6@m9
164: (t4)     |:
165: (t4)     @q8f.e-.@q4d@q8c.d.@q4e-
166: (t4)     @q8f.e-.@q4d@q8c2r@q6{cde-f}
167: (t4)     @q8g.f.@q4_1b-@q8a-.~1g.@q4f
168: (t4)     @q8g.f.@q4e-@q8d.c.>@q4b-<
169: (t4)     :|@q8
170: (t4)     L8>a4r4.gab-<c4r4.>gab-<d4r4.>ab-<de-4r4.>b-<de-
171: (t4)     f4r4.>a-<d-e-f4r4.d-fa-<d-4r4.>fa-<d-f4r4.>a-<d-f
172: (t4)     @q6g-.f.e-d-.>b.<d->b-.a-.g-f.e-.d-e-.f.g-e-.f.b-
173: (t4)     <e-1r2_1{g-edd->b}*80r*2
174:
175: /-------------------------------
176: / CHORD
177:
178: / (t5)と(t6)，(t7)と(t8)，それぞれ似ています。
179:
180: (t5)     L8@4o4@v120p2@k00@q7
181: (t5)     |:7g:||:8a:||:8b-:||:9a:|
182: (t5)     |:7b-:||:8a:||:8b-:||:9a:|
183: (t5)     [do]L8@4o4@v120p2@k00@q7
184: (t5)     |:7g:||:8a:||:8b-:||:9a:|
185: (t5)     |:7b-:||:8a:||:8b-:||:9a:|
186: (t5)     L4@5o4@v116p3@k-4@q8
187: (t5)     |:g.g.gg.g.gg.b-.gg2r4{e-fga-}4
188: (t5)     b-.a-.<e-d-.>b-.a-b-.a-.gg.f.e-:|@q6
189: (t5)     |:re-8f8^2rg8a8^2:||:rd-8f8^2rf8a-8^2:|
190: (t5)     L8@7o3@v105p3@q6
191: (t5)     e-.d-.>b-g-.g-.g-g-.e-.e-d-.d-.>b-
192: (t5)     b-.<d-.e->b-.<d-.g-b-1r2
193: (t5)     r.@8o3q8@v120@k-8(e->g)<(c>e-)<(c16>a-)
194:
195: (t6)     L8@4o4@v120p1@k00@q7
196: (t6)     |:|:7e-:||:8f:||:8g:||:9f:|:|
197: (t6)     [do]L8@4o4@v120p1@k00@q7@m@a
198: (t6)     |:|:7e-:||:8f:||:8g:||:9f:|:|
199: (t6)     L4@5o4@v124p1@k00@q8@m11@a2
200: (t6)     |:g.g.gg.g.gg.b-.gg2r4{e-fga-}4
201: (t6)     b-.a-.<e-d-.>b-.a-b-.a-.gg.f.e-:|@q6
202: (t6)     |:re-8f8^2rg8a8^2:||:rd-8f8^2rf8a-8^2:|
203: (t6)     L8@7o3@v123p1@q6@k5
204: (t6)     e-.d-.>b-g-.g-.g-g-.e-.e-d-.d-.>b-
205: (t6)     b-.<d-.e->b-.<d-.g-b-1r2<_1{dd->bag-e}2
206:
207: (t7)     L8@1o1@v127p2@k04@q14
208: (t7)     |:c1f1|e-1f^2d4d-:|>b-1<c^2cc>b-<
209: (t7)     [do]L8@1o1@v127p2@k04@q14@m@a
210: (t7)     |:c1f1|e-1f^2d4d-:|>b-1<c^2cc>b-<
211: (t7)     L4@5o4@v124p2@k00@q8@m11@a2
212: (t7)     |:b-.b-.b-b-.b-.b-<c.c.c>b-2r4{ga-b-<d-}4
213: (t7)     e-.d-.gf.e-.d-e-.d-.>b-b-.g.g:|@q6
214: (t7)     |:rg8a8^2rb-8<c8^2>:||:rf8a-8^2ra-8<d-8^2>:|
215: (t7)     L8@6o4@v121p2@q6
216: (t7)     b-.a-.g-e-.e-.e-e-.>b-.b-a-.a-.g-
217: (t7)     g-.a-.b-g-.a-.<e-g-1r2_1{bag-edd-}2
218:
219: (t8)     L8@1o1@v126p1@k-3@q14
220: (t8)     |:c1f1|e-1f^2d4d-:|>b-1<c^2cc>b-<
221: (t8)     [do]L8@1o1@v126p1@k-3@q14
222: (t8)     |:c1f1|e-1f^2d4d-:|>b-1<c^2cc>b-<
223: (t8)     L4@5o1@v107p3@k05@q8
224: (t8)     |:b-.b-.b-b-.b-.b-<c.c.c>b-2r4{ga-b-<d-}4
225: (t8)     e-.d-.gf.e-.d-e-.d-.>b-b-.g.g:|@q6
226: (t8)     |:rg8a8^2rb-8<c8^2>:||:rf8a-8^2ra-8<d-8^2>:|
227: (t8)     L8@6o4@v105p3@q6
228: (t8)     b-.a-.g-e-.e-.e-e-.>b-.b-a-.a-.g-
229: (t8)     g-.a-.b-g-.a-.<e-g-1r2
230: (t8)     r@8o3q8@v123@k00(e->g)<|:(c>e-)<:|
231:
232: /-------------------------------
233: / ADPCM RHYTHM
234:
235: /        BASS
236: (t9)     L8o2p3q4@r1
237: (t9)     r*1536
238: (t9)     [do]
239: (t9)     r*1536
240: (t9)     |:8cr4.ccr4cr2crc:|
241: (t9)     |:13cr4.|rcrc:|r2
242:
243: /        SNARE
244: (t10)    L8o4p3q5@r1
245: (t10)    r*1536
246: (t10)    [do]
247: (t10)    r*1536
248: (t10)    |:58r4|dr:|r4
249:
250: /-------------------------------
251:
252: (t1)     [loop]
253: (t2)     [loop]
254: (t3)     [loop]
255: (t4)     [loop]
256: (t5)     [loop]
257: (t6)     [loop]
258: (t7)     [loop]
259: (t8)     [loop]
260: (t9)     [loop]
261: (t10)    [loop]
262:
263: (p)
```

リスト2　MUTE CITYの音色コンフィグファイル

```
1:0000060A 00001BC0   2:0000060B 00001BC0   3:0000060A 00001BC0   4:0000060A 00001BC0
5:0000060B 00001BC0   6:0000060A 00001BC0   7:0000060A 00001BC0   8:0000060C 00001BC0
9:0000060A 00001BC0  10:0000060A 00001BC0
```

リスト3　MUTE CITY用カウンタ表示

```
/ -F･ZERO- MUTE CITY (C)Nintendo  Programed by ENG 93/01/19

/ nomsd.pcm ･･ 創刊X周年記念PRO-68Kより（SION2に
/ 使われていたものです）。スネアドラムをこれに変えても
/ ナカナカいいです。

1=bass1.pcm,v32
2=reals.pcm,v30,p1

/ 2=nomsd.pcm,v57,p-1

.o2c=bass1.pcm,v110,m1,d800
.o4d=reals.pcm,v90,m2,d700,p1
/ .o4d=nomsd.pcm,m2,v142,d700,p-1

.erase 1
.erase 2
```

リスト4　ケンのテーマ

```
 1: /------------------------------------------------------------
 2: /                 STREET FIGHTER 2 AMERICA(KEN)
 3:
 4: /                     composed by ALFH LYRA
 5:
 6: /                       1992 / 1 / 10
 7:
 8: /                     programmed by NAKA
 9: /------------------------------------------------------------
10: .comment STREET FIGHTER 2 AMERICA(KEN) programmed by NAKA
11:
12:                              / ～おことばタイム～
13: (i)                          / スト2の音楽の中ではではこれが一番
14: (b1)                         / イカス（死語）よね！それを今回はバ
15: (o182)                       / リバリ（死語）のロックにアレンジし
16:                              / てみました。アレンジしたというより、
17: (m1,2000)(amidi1,1)          / スト2のアレンジ曲"春雷"の中にちょっ
18: (m2,2000)(amidi2,2)          / とだけ健？(KEN) の曲が入っていたの
19: (m3,2000)(amidi3,3)          / ですが、それがバリバリ（死語）のロッ
20: (m4,2000)(amidi4,4)          / クにアレンジされていて猛烈（死語）
21: (m5,2000)(amidi5,5)          / に気に入ってしまったのです。
22: (m6,2000)(amidi6,6)          / ということで作ってみました。
23: (m7,2000)(amidi7,7)          / アレンジたから嫌だぁーなんて言わない
24: (m8,2000)(amidi10,8)         /   て聴いてみそ！
25: (m9,2000)(amidi10,9)
26:
27: /---- 初期化 -------------------------(ショキカスルナンテンョーキカ！)^_^;
28:
29: .roland_exclusive $10,$42={$40,$00,$7f,$00}
30:
31: /       REVERB&CHORUS SET
32: .roland_exclusive $10,$42={$40,$01,$30
33:                            $03,$03,$00,$4d,$42,$00,$00
34:                            $02}
35:
36: /  PART 4   TONE   SET
37: .roland exclusive $10,$42={$40,$11,$30
```

▶うにうに，ついにサードパーティからSCSIボードが出た。嬉しいかぎりです。これを機に増設RAMもバシバシ出してほしい。それにしてもPOLYPHONはいいな，ぜひほしい。でも，みかか代がつらい今日この頃です。　　　藍原　和久(21)徳島県

```
 38:                          $50,$45,$72,$58
 39:                          $40,$72,$40,$40}
 40: /  PART 5  TONE  SET
 41: .roland_exclusive $10,$42={$40,$15,$30
 42:                          $50,$45,$72,$58
 43:                          $40,$72,$40,$40}
 44: /        RESERVE SET
 45: .sc55_v_reserve $10={1,2,4,3,0,0,0,0,0,4,0,0,0,0,0,0}
 46:
 47: /---- BASS -----------------------------------------------
 48: (t1) r4n1@34@u127v13_2q8@p56o2l8@k-3 @i$41,$10,$42 @e50,20
 49:
 50: f1&f2cfe-r e-4d-rd-1 rd-c>q7b-4|:14b-:|<c4|:14c:||do|
 51: |:f4|:7f :|rf f rf e-4 d-4|:7d-:|rd-d-rd-c4>b-4
 52:     |:7b-:|rb-b-rb-a-b-<c4|:7c :|rc c rcc|c:|r
 53: l4|:|:d-.|d-.d-:|d-d-8d-|>|:b-.|b-.b-:|b-b-8b-8<l8
 54: f4|:7f:|rffe-fe-4r:|
 55: l4|:e-.|e-.e-:|e-e-8e-8 l8c4cfa-<c>b-a-<cr>crcccr
 56: |:f4 |:7f :|rffrfe-4d-4    |:7d-:|rd-d-rd-c4>|
 57:   b-4|:7b-:|rb-b-rb-a-b-<c4|:7c :|rccccr:|
 58:   e-4|:6e-:|e4|:6e:|f&f1&f2..
 59:
 60: /---- MELODY ---------------------------------------------
 61: (t2) r4n2@63@u120v15q8p3o5l8 @i$41,$10,$42@e110,30@h42@m80
 62:
 63: f1&f2cfe-r e-4d-rd-1&d-d-c>b-1&b-2..b-<c&c1&c1|do|l4o5
 64: |:|:3rcfb-8|a-g8f4.e-8r8f8:|<c4d-8c8>|
 65: b-a-8b-8<c2`8>b-a-gf8e2`8:| b-4a-8g8<c8r>b-2a-ga-8g2`8@m90
 66: v11~2@57o4l8|:f4.g^2a-4.b-4a-ge-d-4.<c^2&c2.>b-<c&c+384>:|
 67: ~3@m80@63o5l4rcfb-8<c4.>l8b-a-rgra-4.gfre-rf4 r4fga-2
 68: rb-4a-4gb-rb-a-4g4a-l4g.feg8<c8r8|:5c8:|r8>
 69: <c.>b-a-b-8a-.gf8ca-.gf8e-frf8g8a-2r8b-<c
 70: l8d-crc>b-4a-reg4.f4r2.v13
 71:
 72: /---- BACK -----------------------------------------------
 73: (t3) r4n3@49@u116v12q8@p74o5l8@i$41,$10,$42@e55,10@h42@m60
 74:
 75: 'a-<cf'288'a-<c''a-<cf''gb-<e-'
 76: r'g4b-<e-''fa-<d-'r'f4a-<d-'
 77: r'fa-<d-''e-g<c''d-fb-'360'd-fb-''eg<c'384|do|@49o5v11~2
 78: |:'fa-<c'384'fa-<d-'384'fb-<d-'384|'g<ce'384:|'g<ce'240
 79: @62v14o5|:5'eg<ce':|r4@92o5v11
 80: |:|:c+fa-<c>|r2:|rb-a-g||:>b-<d-fa-|r2:|rra-gr|r1:|
 81: |:e-gb-<e->|r2:|r<d-c>b-r1r2..v10@49o5@u127
 82: |:'fa-<c'384'fa-<d-'384|'fb-<d-'384'g1<ce'
 83: r4@62v14o5|:5'eg<ce':|r@49v11o5:|
 84: ~2'g1b-<e-''g1b-<e''a-<cf'384
 85:
 86: /---- GUITAR ---------------------------------------------
 87: (t4) r4n4@31@u124v10q8@p14o4l8@i$41,$10,$42
 88:  @e75,80@h42@m60
 89:
 90: 'cf'288>'a-<c'<'cf'>'b-<e-'r'b-4<e-''a-<d-'r'a-1<d-'
 91: r'a-<d-''g<c' @q1
 92: |:'f4b-'|:'fb-''b-<f''fb-':|:|
 93: |:'g4<c'|:'g<c''<gc''g<c' :|:||do|o3v10~2
 94: |:|:'f4<c' |:'f<c''<cf''f<c' :|:|o3
 95:    |:'a-4d-'|:'a-d-''d-<a-''a-d-' :|:|
 96:    |:'b-4f' |:'b-f''b-<f''b-f' :|:||o3
 97:    |:'g4<c' |:'g<c''<gc''g<c' :|:|:|
 98:      'g4<c' |:'g<c''<gc''g<c' :|
 99:      rr~5|:5'g<c':|_9r1 o3
100: 'a-d-'384'f>b-'360|:|:'f4<c'|:'f<c''<fc''f<c' :|:|'fc'
101: 'a-d-'384'b-e-'360|:|:'g4c'|:'gc''g<c''gc' :|:|~5
102: |:|:'f4<c' |:'f<c''<cf''f<c'    :|:|o3
103:    |:'a-4d-'|:'a-d-''a-<d-''a-d-':|:||
104:    |:'b-4f' |:'b-f''b-<f''b-f'   :|:|
105:      'g4<c' |:'g<c''<gc''g<c'    :|
106:   q6'g4<c'q7~3|:5'g<c':|r_3        :|@q1
107:   'b-4<e-' |:'b-<e-''<b-e-''b-<e-':|
108:   'b-4<e ' |:'b-<e ''<b-e ''b-<e ':|
109:   o4'cf'192rr @e110,70|:'cf':|r|:'cf':|r@e75,80
110:
111: /---- GUITAR2 --------------------------------------------
112: (t5) r4n5@30@u120v9q8@p84o3l8@i$41,$10,$42 @k5
113:  @e75,80@h42@m60
114:
115: 'cf'288>'a-<c'<'cf'>'b-<e-'r'b-4<e-''a-<d-'r'a-1<d-'
116: r'a-<d-''g<c'
117: |:q8'f4b-'q6|:'fb-''b-<f''fb-':|:|
118: |:q8'g4<c'q6|:'g<c''<gc''g<c' :|:||do|o2v9
119: |:|:q8'f4<c' q6|:'f<c''<cf''f<c' :|:|o2
120:    |:q8'a-4d-'q6|:'a-d-''d-<a-''a-d-' :|:|
121:    |:q8'b-4f' q6|:'b-f''b-<f''b-f' :|:||o2
122:    |:q8'g4<c' q6|:'g<c''<gc''g<c' :|:|:|
123:      q8'g4<c' q6|:'g<c''<gc''g<c' :|
124:      rr~5|:5'g<c':|_9r4 o2q8
125: 'a-d-'384'f>b-'360|:|:q8'f4<c'q6|:'f<c''<fc''f<c':|:|'fc'
126: q8'a-d-'384'b-e-'360|:|:q8'g4c'q6|:'gc''g<c''gc' :|:|~5
127: |:|:q8'f4<c' q6|:'f<c''<cf''f<c'    :|:|o2
128:    |:q8'a-4d-'q6|:'a-d-''a-<d-''a-d-':|:||
129:    |:q8'b-4f' q6|:'b-f''b-<f''b-f'   :|:|
130:      q8'g4<c' q6|:'g<c''<gc''g<c'    :|
131:      q4'g4<c' q6~3|:5'g<c':|r_3      :|
132:   q8'b-4<e-'q6 |:'b-<e-''<b-e-''b-<e-':|
133:   q8'b-4<e 'q6 |:'b-<e ''<b-e ''b-<e ':|
134:   q8o3'cf'192rq6r@e110,70|:'cf':|r|:'cf':|r@e75,80
135:
136: /---- DRUMS ----------------------------------------------
137: (t8) r4n10@17@u115v13 o2l4 @i$41,$10,$42 @e30,20
138:
139: |: crrr8 c8c8rr8c8d8|c:|c8c8l8
140: rcdrccdc rcdrccdr ccdrccdc rcdrccdc|do|
141: |:8rcdrcc|dc:|dc
142: |:7rcdrcc dc:|c8@r1|:5'd<d>':|r4@r014
143: |:|:c.c.dc.|c.@u127<d8>b8@u115:|c@u127<d8>b8@u115c8
144: l8rcdrccdc rcdrccd|d:|cl8
145: rcdrccdc rcdrccdc |:3rcdrccdc|rcdrccdc:|cddd<d>bcc
146: rcdrccdc rcdrccdc |: rcdrccdc rcdrccdc:|l4rccc cc|8ddcc
147:
148: /---- HAT ------------------------------------------------
149: (t9) r4n10 @u127 o2l4
150:
151: |:'<c+a'rrr rrr8<c+8|a>:|l8
152: @u127<a>@u110|:16f+:|@u127<c+>@u110|:15f+:||do|
153: |:4@u127<c+>@u110 |:7f+:|a+|:7f+:|
154:           <a >      ||:7f+:|a+|:7f+:|:||:7f+:|r|:5a+:|r4
155: |:r<c+>f+f+a+4.r|:3 rf+f+f+a+4.r:||:6f+:|a+a+4|:7f+:|:|
156: |: @u127<c+>@u110 |:7f+:|a+|:7f+:|
157:    @u127<a >@u110 |:7f+:|a+|f+|:6f+:|:|f+|:5a+:|f+
158:    @u127<c+>@u110 |:7f+:|a+|:7f+:|
159:    @u127<a >@u110 |:7f+:|a+|:7f+:|
160:    @u127<c+>@u110 |:7f+:|@u127<a>@u110|:7f+:|
161:    @u127<'c+a'>r1r2..
162:
163: /---- LOOP -----------------------------------------------
164:
165: (t1) [loop]/  SC-55はいいですよ！何が良いのかだって？
166: (t2) [loop]/  音が良いんです。あの価格てこの音質が楽しめる
167: (t3) [loop]/  なんて。MO-32なんかよりはずっといいです。
168: (t4) [loop]/  今回のプログラムては本当はFMも鳴らそうとし
169: (t5) [loop]/  てたんだけと、そんな必要はありませんてした。
170: (t8) [loop]/  だって、音が良いんだもん！
171: (t9) [loop]/                          BY  もとMTユーザー
172:
173: (p)
```

リスト5　ケンのテーマ用カウンタ表示

```
1:00000618 00002100   2:00000630 00002100   3:00000618 00002100   4:00000618 00002100
5:00000618 00002100   6:00000000 00000000   7:00000000 00000000   8:00000630 00002100
9:00000618 00002100
```

リスト6　晴れたらいいね


```
10 '
20 '    Dreams Come True
30 '
40 '      「 ﾊﾚﾀﾗ ｲｲﾈ 」
50 '
60 ' Programed by T.abe 1992
70 '
80 DIM a$(20)
90 GOSUB 2430
100 PLAY 0:IF f THEN PLAY "x"
110 GOTO 170
120 '
130 LABEL"P"
140 READ a:IF a=255 THEN PLAY ":";: RETURN
150 PLAY a$(a);:GOTO 140
160 '
170 ' Vocal 1
180 a$(0)="i7o4v16L12k0p3q7"
190 a$(1)="e4q6d6q7cd6e&e6<q6gq7>e4q6d6q7cd6e&e4 e4q6d6cd6ef6eq5
g6q7d&e6c&c6<a>cde"
200 a$(2)="d4q6c6q8<a&a6>c&c4&c4 r4r6<a>cde- d4q6c6q7<a-&a-6>c&c
4 r2.r6<g>"
210 a$(3)="e4q6d6q8c12d6e4d&e4 q6d6q7cd6e4q8de4q6d6q7cd6ef6eq5g6
q7e4c&c4cde"
220 a$(4)="d4q6c6q8<a4>c&c4 r2.q7cde-d4q6c6q8<a-4>c&c4 d2e-2"
230 a$(5)="g@36&f+@2&f@4&f+@2&g@52r6q6fq7e-6cd6d4q8c&<b-4r4> g4r
4r6q6fe-6cq8e-2e2"
240 a$(6)="a@36&g+@2&g@4&g+@2&a@52r6q6gq7f6de6q8e4d&c4r4 d4&dcde
4&edef1 r4i4s4,1,0,20=3>a4<=0i7q7cdffga"
250 a$(7)="q8>c2<f4r6q5f>q8c6q7c6c6q8c6<b-6q7a6q8b-4&b-6fg6&f&f4
r1"
260 a$(8)="q8>c2<f4r6q5f>q8c6q7c6c6q8c6<b-6q7a6q8b-6&b-4ff4a6g&g
2r2"
270 a$(9)="r1"
280 a$(10)="q8b-6&b-4ff2"
290 a$(11)="q8>d4e4f4g4a2b-2>c6<<r=3i4>a4<=0i7q7cdffga"
300 a$(12)=">q8c6&c4q7d<q8f4r6q6f>q8c6q7c6c6q8c6<b-6a6"
310 a$(13)=">q8c6&c4<q7fq8f4r6q6f>q8c6q7c6c6q8c6<b-6a6"
320 a$(14)=">q8c6&c4<q7fq8f4r6q6f>q8c6q7c6c6q6c6q7d6c6<"
330 a$(15)="q8a6&a4fg6&f&f4r2.f4"
340 a$(16)="q8b-6&b-4ff4a6g&g2r4r6"
350 a$(17)="q6cq8b-b-ab-6a&a4r6q6c q8b-b-ab-6a&a4 r6q6aq8ab-ab-6
a&a2"
360 a$(18)="i26k5p1v15o3g1&g1L12 r4b6b>c6cd6d"
370 a$(19)="i18v12o4r1r1r4e6f&f2 r4g6a&a2&a1&a2"
380 DATA 18,0,1,2,3,4,5,6,7,13,10,9
390 DATA 3,4,5,6,7,12,10
400 DATA 9,9,9,9, 9,9,11, 8,14,15,13,16 ,17 ,19,255
```

▶センター試験を受けてきました。そこで，マークシートである解答用紙が配られるたび，解答番号を使って裸眼立体視をしていたので目が疲れてしまいました。

石井　大輔(18)東京都

```
410 GOSUB "P"
420 ' Vocal 2
430 a$(0)="i7o4v14L12k5p3"
440 a$(3)="g4f6e12f6g4f&g4 f6ef6g4fg4f6ef6ga6g>c6<a4f&f4cde"
450 a$(6)="a2r6gf6de6e4d&c4r4 f4&fefg4&gfga1 r4i4s4,1,0,20=3>a4<
=0i7cdffga"
460 a$(11)="b-4>c4d4e4f2g2a6<r=3i4>a4<=0i7cdffga"
470 a$(17)="fagfg6f&f4r6fagfg6f&f4r6fagfg6f&f2"
480 a$(18)="i18k5p2v12o4r4f2.&f1L12r4<g6ga6ab6b"
490 a$(19)="i18v11o3c2.r6<g> c2.r6<g> c2.r6<g> c1&c1&c2"
500 RESTORE 380
510 GOSUB "P"
520 ' Piano 1
530 a$(0)="i10o4v13L4q7p3"
540 a$(1)="gggg"
550 a$(2)-"ffff"
560 a$(3)="aaaa"
570 a$(4)="a-a-a-a-"
580 a$(5)="a6a4a12&a2"
590 a$(6)="ffgg"
600 a$(7)="ggg+g+"
610 a$(8)="ggaa"
620 a$(9)="b-6r12r2."
630 a$(10)="r6f12f4f6f12&f4"
640 a$(11)="i26k5p3v15r2o4d2&d1L12r1"
650 a$(12)="r4c6d12&d2"
660 a$(13)="&d1&d2"
670 a$(14)="L8a2rar4 b-2rb-r4"
680 a$(15)="b-rb-r4"
690 DATA 11,0,1,1,1,2, 2,3,4,5, 1,1,1,2, 2,3,4,6
700 DATA 1,1,1,7, 3,3,8,3,9, 3,3,1,10,3,3,1,9
710 DATA 1,1,1,2, 2,3,4,6, 1,1,1,7, 3,3,8,3,9
720 DATA 3,3,1,10, 3,3,1, 1,1,1,1, 3,3,8,3,9
730 DATA 3,3,1,10, 3,3,3,9, 3,3,1,10, 3,3,1
740 DATA 12,12,12,12,13, 14,14,15,14,14,15,14,14,255
750 GOSUB "P"
760 ' Piano 2
770 a$(0)="i10o4v13L4q7p3"
780 a$(1)="eeee"
790 a$(2)="cccc"
800 a$(3)="dddd"
810 a$(4)="e-e-e-e-"
820 a$(5)="e6e4e12&e2"
830 a$(6)="dde-e-"
840 a$(7)="e-e-ee"
850 a$(8)="ffff"
860 a$(9)="ddee"
870 a$(10)="g6r12r2."
880 a$(11)="r6d12d4d6d12&d4"
890 a$(12)="f6r12r2."
900 a$(13)="i18k5p2v12o4r2.a4&a1r1"
910 a$="r4a6&b-12&b-2"
920 a$(14)="<"+a$+a$+a$+a$+"&b-1&b-2>"
930 a$(15)="L8 e2rer4 f2rfr4"
940 a$(16)="frfr"
950 DATA 13,0, 1,1,1,2, 3,1,4,5, 1,1,1,2, 3,1,4,6
960 DATA 3,3,4,7, 8,1,9,8,10, 8,1,3,11, 8,1,3,12
970 DATA 1,1,1,2, 3,1,4,6, 3,3,4,7, 8,1,9,8,10
980 DATA 8,1,3,11,8,1,3, 4,3,4,9,8,1,9,8,10
990 DATA 8,1,3,11, 8,1,8,12, 8,1,3,11, 8,1,3, 14
1000 DATA 15,15,16,15,15,16,15,15,255
1010 GOSUB "P"
1020 ' Piano 3
1030 a$(0)="i10o4v13L4q7p3"
1040 a$(1)="cccc"
1050 a$(2)="<bbbb>"
1060 a$(3)="<b-b-b-b->"
1070 a$(4)="<aaaa>
1080 a$(5)="<a6a4a12&a2>"
1090 a$(6)="<aa>cc"
1100 a$(7)="<b-b-bb>"
1110 a$(8)="<b-b->cc"
1120 a$(9)="dddd"
1130 a$(10)="d6r12r2."
1140 a$(11)="<r6b-12b-4b-6b-12&b-4>"
1150 a$(12)="<b-6r12r2.>"
1160 a$(13)="i26k5v15r1o5c1r1"
1170 a$="r4f6f12&f2"
1180 a$(14)=a$+a$+a$+a$+"&f1&f2>"
1190 a$(15)="L8c2rcr4 d2rdr4"
1200 a$(16)="drdr"
1210 DATA 13,0,1,2,3,4, 4,1,1,5, 1,2,3,4, 4,1,1,6
1220 DATA 3,3,1,7, 1,1,8,9,10, 1,1,3,11, 1,1,3,12
1230 DATA 1,2,3,4, 4,1,1,6, 3,3,1,7, 1,1,8,9,10
1240 DATA 1,1,3,11, 1,1,3, 3,3,1,7, 1,1,8,9,10
1250 DATA 1,1,3,11, 1,1,9,10, 1,1,3,11,1,1,3 ,14
1260 DATA 15,15,16,15,15,16,15,15,255
1270 GOSUB "P"
1280 ' Brass
1290 a$(0)="i18o4v12L12q7p3"
1300 a$(1)="r1r1r1r1"
1310 a$(2)="r6e-d6<b->e-6fg6a-b-1r1r1"
1320 a$(3)="r6af6cf6ga6b->c1r1 r4ddcddc<a6r r2cdffga"
1330 a$(4)="r4f6fd6c<a6ar1r1r6>ff4f6f&f4 r4f6fd6c<a6ar1r1>"
1340 a$(5)="r1g4f6ef6gr6fg4f6e<ggg>c6d<b-6rb-6rb-6rb-6ra6aa2."
1350 a$(6)="r2.a4>e6eeeec2 e-2r2<a6ra6r>c6rc6r"
1360 a$(7)="r6e-d6<b->e-6fg6a-b-1r1r1"
1370 a$(8)="v16i26f@1&f+@1&g@30 rr4r6fe-6cd6dr6<b-&b-2> f@1&f+@1
&g@30 rr4r6fe-6ce-2e2"
1380 a$(9)="g@1&g+@1&a@30 rr4r6gf6de6er6c&c2> i18v12r4ddcr4ddcr4
ddcddc<a6rr2cdffga"
1390 a$(10)="r1r4f6fd6c<a6ar2.>g6f&f1 r6ff4f6f&f4"
1400 a$(11)="r4f6fd6c<a6ac2r2r1"
1410 a$(12)="r16k5p1i7v14r1L24o5>ecec<ag>c<a>c<agfagagfdgfgfde&e
2.&e8."
1420 a$(13)="q8o4r4e6f&f2 r4e6f&f2 r4g6a&a2 r4b-6>c&c2&c1&c2"
1430 a$(14)="L8<argfgara r4gfgara r4gfgar>c <rarfrfgf>r2"
1440 DATA 12,0,1,1,1,1,2,3,4,5,6,7,3,4,8,9,4,10,11 ,13,14,14,14,
255
1450 GOSUB "P"
1460 ' Bass
1470 a$(0)="i31o3L12v13q7p3s4,1,0,20=0"
1480 a$(1)="c6rc6<g>c6rc6<g b6rb6gb6rb6g b-6rb-6fb-6rb-6f f6rf6>
c<f6rf6>c"
1490 a$(2)="d6rd6<aa6ra6a"
1500 a$(3)="<f6fr6ff4>dfg"
1510 a$(4)="a6ra6ea6ra6e a-6ra-6>e-<a-6ra-6>e-"
1520 a$(5)="d6rd6<a>d6rd6<a"
1530 a$(6)="d6r<b-6ra-6rf6r>"
1540 a$(7)="e-6re-6<b->e-6re-6<b- g6rg6>d<g6rg6>d"
1550 a$(8)="c6rc6<g>c6rc6<g> e-6re-6<b->e6rc6r"
1560 a$(9)="<f6rf6>cc6rc6<g a6ra6ea6ra6e g6rg6ga6ra6a g6rg6>dg6d
g6dc6rr2 =3c4=0"
1570 a$(10)="<f6rf6>c<f6rf6ga6ra6ea6ra6ag6rg6>d<g6rg6>d"
1580 a$(11)="r6cc4c6c&c4"
1590 a$(12)="c6rr2."
1600 a$(13)="e-6re-6<b->e-6re-6<b- g6rg6>d<g6rg6>d c6rc6<g>c6rc6
<g> e-6re-6<b->e6re6r"
1610 a$(14)="<f6rf6>c<f6rf6g a6ra6ea6ra6a g6rg6ga6ra6a g6rg6>dg6
dg6dc6rr2."
1620 a$(15)="<f6rf6>c<f6rf6ga6ra6ea6ra6ag6rg6>dr6d<g4>"
1630 a$(16)="i31k5v15r1o2g32&g+32&a8.&a2. L12g6rg6rg6rg6r"
1640 a$="c2.r6<g>"
1650 a$(17)=a$+a$+a$+"c1&c1&c2"
1660 a$(18)="L8<f4>rc<f4>rc <f4>rc<f4>rc <f4>rc<f4b->cc4rcc4r<g>
crcr"
1670 RESTORE 1680
1680 DATA 16,0,1,2,4,3,1,5,4, 6,7,8,9,10,11,10,12
1690 DATA 1,5,4,6,7,8,9,10,11,10
1700 DATA 13,14,10,11,15,12,10,11,10, 17,18,18,18,255
1710 GOSUB "P"
1720 ' Drums
1730 a$(0)="v16s4,1,0,30=3L12o2q8p3"
1740 a$(2)="i3e4i2e6i3e"
1750 a$(1)=a$(2)+a$(2)
1760 a$(3)="e4i2eee"
1770 a$(4)="e4i2e6e"
1780 a$(5)="i2e4i3e4i4aafd6c"
1790 a$(6)="i2e4i3e4e4i4afc"
1800 a$(7)="i3e6i4af6c"
1810 a$(8)="i4aaaf6c"
1820 a$(9)="i2e4i3e4e6i4af4"
1830 a$(10)="i2e4r6i3ei4aaaf6d"
1840 a$(11)="i4a16f8.i3e4"
1850 a$(12)="t100r1r1o2v16t121s4,1,0,20=3i2e4i3e4e4i2e12e12i4a12
"
1860 a$(13)="i3e4i2e6er2t100r2r2t140L8i4afi2e16e16e"
1870 a$(14)="i3e4i2ei3ee4i2ei3e"
1880 a$(15)="i3e4e4"
1890 DATA 12,0, 1,1,1,1, 1,1,1,2,3, 1,1,1,1, 1,1,1,2,4
1900 DATA 1,1,1,2,3, 1,1,1,1,5, 1,1,1,2,4, 1,1,2,3,6
1910 DATA 1,1,1,1, 1,1,1,2,7, 1,1,1,2,8, 1,1,1,1,9
1920 DATA  1,1,1,2,3, 1,1,2,8, 2,4,2,4,2,4,2,3
1930 DATA 1,2,4,2,4,2,4,10, 1,1,1,2,3, 1,1,1,11,3, 1,1,1,2,3, 1,
1,2,4
1940 DATA 1,1,1,13, 14,14,14,14,15,14,14,14,14,15,14,14,14,14, 2
55
1950 GOSUB "P"
1960 ' PSG 1
1970 a$(0)="s2,1,0,1h4=1v13L12o6k0"
1980 a$(1)="r1L24v15k0o5^2>ecec<ag>c<a>c<agfagagfdgfgfde&e1"
1990 a$(2)="r1L12o6c<afafdfd<b>d<bg"
2000 a$(3)="r1r1"
2010 a$(4)="^0e1&e4e4f4g4g2.g6a>c6<af6a&a4>c6r ^2c6r<a6r>c6r<a6r
> c4<a6ga6>c4r c6r<a-6r>c6r<a-6r f6rf6rg6rg6r"
2020 a$(5)="b-1r1r1g6rg6rg+6rg+6r a1r2r6>ec6<a r4>ddcr4ddcr4ddcd
dc<a6r r1>"
2030 a$(6)="^0r4a4e4f4e4a4e4e4"
2040 a$(7)="f1^2f6<b->f6<b->f6<b->f6<b->"
2050 a$(8)="f2a2r4a4g4f4<"
2060 a$(9)="f1^2fd<b->fd<b->fd<b->fd<b->"
2070 a$(10)="f6&f4ga2"
2080 a$(11)="v12g1a-2b-2g1 >^2fd<b->d<b-f>g-e-<b>e-<b-g- ^0a1b-2
>c2<v13^2r4b-b-ar4b-b-ar4b-b-ab-b-af6rr1"
2090 a$(12)="r4a4g4f4"
2100 a$(13)="r4f4g4a4e4a4e4g4"
2110 a$(14)="^0r4a4g4f4e4d4c4e4f6&f4ga2"
2120 a$="r4c6d&d2"
2130 a$(15)="^2"+a$+a$+a$+a$+"efegec<a@24g@24&g4r2>"
2140 a$(16)="L8c4cr4cr4 d4dr4dr4 e4eeeere<b-4>d<b-&b-2>drdr"
2150 DATA 1,0,3,3,3,2,4,5,6,7,6,8, 4,5,6,9,6,10,11
2160 DATA 6,9,6,10,12, 13,9,14,15,16, 255
2170 GOSUB "P"
2180 ' PSG 2
2190 a$(0)="s2,1,0,1h4=1v13L12o6k1"
2200 a$(1)="r1L24v15k1o5^2>ecec<ag>c<a>c<agfagagfdgfgfdc&c1"
2210 a$(4)="k2"+a$(4)
2220 a$(6)="k1^0r4f4c4d4c4e4c4c4"
2230 a$(7)="d1^2d6<g>d6<g>d6<g>d6<g>"
2240 a$(9)="d1^2d<b-g>d<b-g>d<b-g>d<b-g>"
2250 a$(10)="d6&d4ef2"
2260 a$(11)="v12e-1f2g2e-1 >^2fd<b->d<b-f>g-e-<b>e-<b-g- ^0f1g2a
2 v13^2r4b-b-ar4b-b-ar4b-b-ab-b-af6rr1"
2270 a$(12)="r4f4e4d4"
2280 a$(13)="r4d4e4f4c4e4c4d4"
2290 a$(14)="^0r4f4e4d4c4<b-4a4>c4 d6&d4ef2"
2300 a$="r4a6&b-&b-2"
2310 a$(15)="^2<"+a$+a$+a$+a$+">r@2efegec<a@24g4&g@22r2"
2320 a$(16)="L8a4ar4ar4 b-4b-r4b-r4> c4ccccrc<f4>d<b-&b-2 b-rb-r
"
2330 DATA 0,1,3,3,3,2,4,5,6,7,6,8, 4,5,6,9,6,10,11
2340 DATA 6,9,6,10,12, 13,9,14,15,16, 255
2350 GOSUB "P"
2360 ' PSG 3
2370 a$(1)="r8 r1L24k1o5^2>ecec<ag>c<a>c<agfagagfdgfgfde&e2.&e8"
2380 DATA 1,255
2390 GOSUB "P"
2400 PLAY ""
2410 END
2420 ' Voice
2430 MEM$(&HB190,36)=HEXCHR$("FA 50 61 08 51 02 1B 27 25 00 5D 5
D 5D 50 04 0F 1F 1F 00 00 00 00 16 58 03 0A 00 80 80 00 00 C8 94
```

```
 00 02 00")'LEED
2440 MEM$(&HB1B4,36)=HEXCHR$("FC 00 0F 02 02 03 00 00 0A 00 1F 1
F 1F 1F 00 12 15 0E C0 0E 00 00 01 18 F9 F8 00 00 00 00 00 C8 94
 00 02 00")'SNEAR DRUM
2450 MEM$(&HB1D8,36)=HEXCHR$("FC 21 0F 0F 00 30 00 04 0A 00 1F 1
F 1F 1F 00 14 0F 08 C0 CC 00 00 08 A8 F8 F8 00 00 00 00 00 C8 82
 02 02 00")'Bass  DRUM
2460 MEM$(&HB1FC,36)=HEXCHR$("FB 00 05 00 07 02 00 14 0A 00 1F 1
F 1F 1F 1F 17 18 00 40 00 8C 0C 03 95 F8 07 00 80 00 00 F4 C8 80
 00 02 00")'Syn.Tom
2470 MEM$(&HB268,36)=HEXCHR$("BA 50 01 03 01 01 1A 2D 1E 00 1F 1
7 17 15 0E 0C 00 00 00 00 00 00 02 F2 02 07 00 00 00 00 D0 C8 94
 00 02 00")'PW MAIN
2480 MEM$(&HB2D4,36)=HEXCHR$("DA 31 71 0D 33 01 23 2D 26 00 5F 9
9 5F 94 05 05 05 07 02 02 02 02 11 11 11 A6 00 00 00 00 00 C8 82
 02 02 00")'E Piano  2
2490 MEM$(&HB3F4,36)=HEXCHR$("FD 00 01 31 11 31 1E 08 0A 08 11 1
4 14 14 07 08 08 08 00 00 07 00 05 0A 0A 0A 00 00 00 00 00 00 80
 00 00 00")'Brass
2500 MEM$(&HB514,36)=HEXCHR$("FC 00 01 11 21 61 1E 00 1C 07 52 1
2 4F 14 00 8A 00 82 01 01 01 01 2A 3A 5A 3A 00 00 00 00 00 C8 80
 00 02 00")'Brass    1
2510 MEM$(&HB5C8,36)=HEXCHR$("C8 00 71 13 71 01 32 2D 19 00 55 5
F 95 1F 0A 87 05 81 0F 0F 0F 0F F4 38 F4 F8 00 00 80 00 F4 C8 80
 00 02 00")'Wood Bass
2520 RETURN
```

リスト7　ムーンライト伝説（1月号に掲載）用カウンタ表示

```
 1:00004A76 00000000   2:00004A76 00000000   3:00004A8C 00000000   4:00004A76 00000000
 6:00004A76 00000000   7:00004A76 00000000   8:00004A76 00000000   9:00004A76 00000000
20:00004A5C 00000000  21:00004A5E 00000000  22:00004A57 00000000  30:00004A74 00000000
31:00004A74 00000000  32:0000493E 00000000  40:00004A5E 00000000  41:000046E6 00000000
50:00004A74 00000000  60:00004A72 00000000  61:00004A6F 00000000  62:00004A6F 00000000
70:00004866 00000000
```

(善)のゲームミュージックでバビンチョ

●ウィンビーのネオシネマ倶楽部
——エバーグリーン編——　CD：KICA-7612
キングレコード　3,000円(税込)　発売中

ウィンビー国民的アイドル化計画第1弾と銘打って登場したこのCD，これもまた，以前(1月号)に紹介した「コナミ・オールスターズ」のさゆ鈴とかのぶっとんだノリの企画なのだろうと思っていたが，聴いてみてビックリ，内容はいたってまともであった。

アーケード「出たな!! ツインビー」，ファミコン「MA・DA・RA」，MSX「F1スピリット」などの有名どころのVGMをJAZZ風，ピアノやストリングスの室内楽風のおだやかなアレンジで収録している。勉強やドライブのBGMにはいいかも。

お勧め度　8

●パーフェクトセレクション
サウンド・レーシング・ヒストリー
CD：KICA-1119
キングレコード　3,000円(税込)　2/24発売

コナミがいままでに発売した，MSXからアーケードまでのレーシングゲームのBGMをフュージョン風にアレンジして復刻。

収録タイトルは「F1スピリット(MSX)」「ホットチェイス(アーケード)」など。毎回期待ハズレをやらかしてくれるNazo²なので期待していなかったが，今回のは実にスマートで聴き心地のいいサウンドに仕上がっている。スタンダードなフュージョンのほかにヘビメタ風あり，ブラック風あり，プログレ風あり，となかなか聴き飽きさせない構成。選曲もいい。ギターはもちろんキーボード，ピアノ，オルガン，ソプラノサックスまでもがリードをとったりして音的にも楽しめる。もしかしたらNazo²プロジェクトではいちばんの完成度かも。ところでこのNazo²プロジェクト，このアルバムをもって終了するらしい。う〜む。意味深。

お勧め度　9

●餓狼伝説2/SNK　新世界楽曲雑技団
CD：PCCB-00111
ポニーキャニオン　1,500円(税込)　2/19発売

NEOGEOの100メガショックシリーズの第2弾「餓狼伝説2」のアレンジバージョン1曲を収録したオリジナルサウンドアルバム。前作は個性的な曲調と音素材でVGMフリークを驚かせてくれたが今回はどうか。

結論からいうと，曲，サウンドともに前作よりパワーアップしている。世界中の格闘者が出演キャラクターということで，ナショナリズムあふれる(?)曲調が多いが，ゲームミュージックならではのアレンジがきいていておもしろい。原住民がスティールドラムを叩いて叫びまくる曲には恐怖とも感動ともいえぬ奇妙な気持ちに陥った。それと，この手のアルバムで定番ともいえるボイスコレクションも収録している。その筋の方は大喜びか。

お勧め度　7

●ビューポイント/サミー　CD：PCCB-00112
ポニーキャニオン　1,500円(税込)　2/19発売

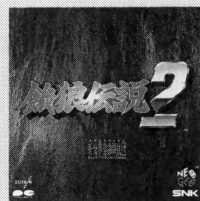
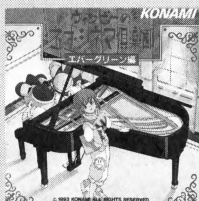
いきなり1曲目のアレンジバージョンのダークで深みのあるハウスミュージックに度肝を抜かれた。これはイイ。耳を突くようなバスドラムと軽いスネアで構成されたタメのきいたリズムパートと，こもり気味のベースに我が道を行くピアノバック，この上をリードオルガンがアドリブで走る走る。ゲームミュージックのアレンジバージョンでは久々のヒットではないだろうか？　いっぽう，オリジナルサウンドのほうも，ワウワウギターとか，あやしい外人のシャウトなどのサンプリングをミックスした個性的ハウスサウンドに仕上がっている。

お勧め度　8

●ドギューーン!!/東亜プラン　CD：PCCB-00113
ポニーキャニオン　1,500円(税込)　2/19発売

シューティングゲームフリークの神様こと東亜プランが放った最新シューティングゲーム「ドギューーン!!」のオリジナルサウンドアルバム。作曲者自らが手がけたアレンジバージョンも1曲収録。もちろん効果音集も付いてるぞ。東亜プランマニア，ドギューンマニアはもう買うしかないだろう。うむ。

お勧め度　6

●サイトロンビデオゲームミュージック年鑑1992
CD(2枚組)：PCCB-00109
ポニーキャニオン　3,500円(税込)　2/19発売

月刊ゲーメストの年間ベストVGM賞より上位5タイトルを収録した2枚組140分のベスト盤。

収録タイトルはタイトー「メタル・ブラック」「ギャラクティック・ストーム」，カプコン「ストリートファイターII'」，SNK「龍虎の拳」，データイースト「ウルフ・ファング」の5タイトル。メタルブラックとストリートファイターII'(あやしいボーカルアレンジ！)のアレンジバージョンは新録だ。去年は1枚もゲームミュージックCDを買わなかったアナタにオススメっていう感じかな。うむ。

お勧め度　6

終わりに

埼玉県の野崎徹君からのレポート。

「突然ですが，「ひょっこりひょうたん島——海賊キッドの宝の巻——」(CD：BCCM-0016/バンダイ/2,800円(税込))のCDを買ってきて聴いてみました。一見，妙にあかぬけていて楽しげですが，男女平等問題，教育問題，飢餓問題などを扱った楽曲もあり，平和に溺れ浸りきった若者たちにぜひオススメです。ところで私はサンデー先生のファンです」

私も買ってきて聴いてみたが，なんだかシブイ内容だな。声優に名古屋章とか若山弦蔵とかがいるぞ，おい，スゲー取り合わせだな，いまじゃ絶対実現しないと思うぞ。さて，私にとって不可解なのは物語途中であんなにサンデー先生を慕っていた子供たちが，突然なぜなんのためらいもなく海賊大学に入ったかってことだ。確かに「深い」内容だ。

それにしても，アホウドリのホウスケ33世のテーマにはヤラレた。「アホアホー」なんつーバックコーラス入れるなんて天才音楽家しか思いつかないよ，ホント，冗談抜きで。

ま，特にオススメはしないけど，興味のある人は聴いてみてくれ。んでは，また来月。

クルマなしVSパソコンなし

Takazawa Kyoko
高沢 恭子

クルマやパソコン。「便利」とか「必需品」なんていわれて，なんとなく購入する人も。でも，ちょっと待って。あなたにとってなにが「便利」で「必要」か，キョウコさんと考えてみませんか。

電子工学を学んでいる大学3年生の甥，遠藤真樹(まさき)君が，自宅にパソコンを買いたいが，どの機種がいいだろうかと相談の電話をしてきた。

夫が電話口に出て，すこし話をしてみたけれど，こういう話は縁談と同じくらいむずかしいものだ。

どんな特徴を持ったパソコンがいいのか，おもにどういうことをしたいのか，ふつうはそんなことを焦点に決めていくもので，そのほうが候補もあげやすい。

だが真樹君の場合，コンピュータは専門分野の一部であり，現状では実用というより研究素材に近いようだ。なにに使うという目的がはっきりしているわけではないらしい。そうなると，もうどれがいいかなんてわからなくなる。

「電話でなく，またよく話しましょう」

こちらもじつは真樹君のお父さんに用事があって，新年のあいさつがてら横浜の家に夫婦で出かけることになっていた。

こわーい質問

少女漫画のヒロインよりずっと可愛かったトコちゃんは，国立の女子大に通っていた。同い年の東大生と恋をして，卒業後結婚，ただし20年以上前のことだ。長男の真樹君はいまお父さんの後輩なのだ。トコちゃんは私の実妹である。

「パソコンってのは，あれ，家庭ではなにに使えます？」

横浜の遠藤家の応接間で，真樹君のお父さんの庸生(つねお)さんが，とてもあらたまった口調で夫にたずねた。

私は「アッ」と思った。

そんな重大な質問をしないでほしい。そういう心境だった。マニア道も20年に近い夫はどんなことを思っていたか，しばし答えをためらっている。

庸生さん自身も，会社ではたくさんのコンピュータに囲まれて，国産車の設計などを手がけている人である。

夫がすこし考えていると，庸生さんがいった。

「会社でもボクはあまり端末はいじりませんが，たまに使うのはやっぱりワープロとしてですよね。あとは，通信なんかできるのも便利だとは思いますけど」

そう，このあたりの意見が，いまのパソコンに対しての一般的な感想だと思う。

じつはもう真樹君は，このときにはパソコンを買うのは見送るという結論を出していて，この日は用事で外出していた。応接間にはトコちゃん夫妻と，もうひとりの子供，次男で高3の真琴君がいた。

パソコンの購入を見送ったのは，その必要性をあらためて考えなおしたことにあったようだ。準備のつもりでまず費用を検討してみたら，パソコンだけでなく，周辺の機器もあれこれそろえなければならないことに気づいた。それをはじめて知ったわけでもないだろうけれど，満足できる環境を整えるとなると，けっこうな支出になる。

そこであらためて，いまどうしても真樹君にとってパソコンが必要不可欠なものかを考えてみたそうだ。

大学ではいつも授業や研究で，ワークステーションを自由に利用している。あれだけ便利でなんでもできるものを使っていると，自宅にパソコンを買っても，ものたりないような気もしたという。

学校に行けばなんでもそろっていて，使いたい放題だし，いま帰宅後もひきつづき研究しなければならないほど，時間に追われているわけでもない。

お父さんのほうも，家にパソコンを購入したとして，なにかたいへん便利で役だつことがあるだろうかと考えたようだ。その結果，これという画期的な利用法もみつからなかったので，遠藤家では今回の購入を見送ったということらしい。

コンピュータのはたらきはじゅうぶん知っている。では，パソコンがふつうの家庭の中でどういう役にたつか。

この根本的な質問への回答はやさしいものではない。ことに，自分のパソコンを持たない人から問われたらなおさらだ。それは，たとえばいまわが家でどんなことに使っているかというような，項目をあげてみることではダメだろう。質問する人は，パソコンでなければならない，しかも日常的なしごとの強力な実例がほしいのだ。

Oh! march

この日のこちらからの用件は，乗用車を購入しようと思うので，どれがよいか推薦してもらうことだった。

航空学科で学んだ庸生さんは，日産自動車に勤務して20余年。話題になったシーマを設計したメンバーのひとりでもある。

クルマとわが家は，およそ無縁という歳月が長かった。クルマを持たなかったいちばんの理由は，ずいぶん前に，夫が取得しかけた免許を多忙を理由に中止したことだと思う。

小さなきっかけが，方針や価値観を都合よく修正させてしまい，クルマの利用は忘れられたのだ。でも，もしクルマがどうしても必要なら，ふたたびそのための計画をしただろう。そのままですごしてきたのは，なくても困らなかったからだ。また，そうした生活習慣がクルマへの関心をなくさせてしまったこともあると思う。

夫は昨夏から勤務地が近畿に移り，あいかわらず東京との二重生活をしている。

芭蕉生誕の地，伊賀忍者の里，上野城などで名高い三重県上野市だ。まわりにも歴史上ゆかりの名所が多く，奈良，京都も近い。昨年はFBIの「カー・バイク」ボードの主催で，夫の勤務地をめざすかたちの「伊賀ツーリング」も実施された。

ところが，のどかで空気のきれいな街とはいえ，マンションを一歩出るととてもクルマなしの行動は不便とわかった。

ほんとうのところ，夫はこれまで，クルマは自分で持たなくてもなんとかなることが多かった。社用なら送り迎えもしてくれるし，これからもそうかもしれない。でも，自分のためのクルマを持とうと決めた。つまり，パーソナルのクルマを持って，自分がオーナーになることにしたのだ。

無理の多いスケジュールの中で，半年かけて免許を取得した。必要だからできた。

クルマを買う段階が近づくにつれて，まわりのクルマ通が，いろいろとアドバイスをしてくれるようになった。乗用車の車種となると，とてもパソコンの比ではない。みんなそれなりにキャリアのあるユーザーだから，耳をかたむけるだけの説得力があるが，客観的な評価はわかりかねる。

このさい，やはり専門家の庸生さんに聞いてみるのが最善だ。車種も日産のものときめて，アドバイスしてもらおうということになった。

利用の状況としては，だいたいはひとりで乗る場合が多いこと。軽快で身動きがラクなものが希望であること。そんな条件にあわせて，いくつかの候補が出た。

いろいろ話すうちに，夫が自分でも考えていたサニーが最有力になった。

ところが，だまって話を聞いていた真琴君が思いがけない発言をした。

「ありふれたセダンより，マーチみたいな小型車でひとりで走るほうが，紳士らしくてシブいと思うけどなぁ」

真琴君は受験の年なのに，昨年はちゃっかりバイクの免許を取って，おこづかいで中古のバイクも買ってしまったそうで，母親のトコちゃんは困り顔なのだ。

それにしても若い真琴君が，カッコよさを売るハデな車でなく，走りやすさとシブさをあげて推薦したことに，私たちはとても意外に思い，強く動かされた。

それには庸生さんも同感だといった。

「マーチはいいですよ。走りやすい。いまけっこうこれには力を入れてますからね」

夫がふたたび考え出したようすに，

「ちょっと，待っててください」

と，真琴君は2階の部屋までいって車の月刊誌を持ってきた。

「兄貴の本ですけど，ちょうどマーチの特集がありますから」

私も知らなかったのだが，兄弟2人ともクルマが大好きで，とにかくいろいろよく知っていると両親がそばからいう。

夫は数分でマーチに決めてしまった。

自分で下調べをした範囲でもマーチの特徴はなかなかいいと思っていたが，小型の点でためらっていたようだ。それが若い真琴君のひと言のキキメで，即決となった。すくなくとも，その場ではそういうなりゆきだった。でも，ともかくいまは，1台クルマを使ってみること。それが目的らしい。

これがパーソナル

パソコン，クルマ，それぞれ経験の長いどうしで，今回ははからずもお互いにアドバイスをするようなことになった。

パソコンのほうは，詳細な話になる前にもとめる側で見送りの結論を出してしまったので，こちらとしても家庭での利用法についてあえてなにもいうことなく終わった。

わが家での実例をあげるなら，長い間のことだから，実用としてずいぶん大きな役割をしていることも事実なのだ。

電話番号のリストは5年くらい前から大活躍。50音順，局番をふくめた番号順，交際のジャンル別など，わかりやすい。

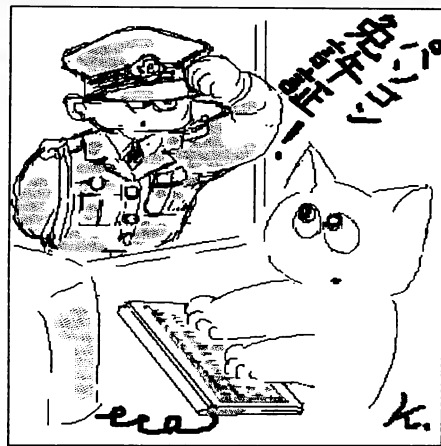
名刺のリストもある。50音順，受けた日付順，ジャンル別など。

年賀状の送受状況，とくに前年に受けた日付とこちらが出した日付を対照することで，出すか否かの参考にする。

コレステロールの計算(食品の摂取量に対する算出)，ビタミン，ミネラルの摂取量の計算など，栄養面での利用。

銀行，信託，証券などの，預金の満期や推移のリスト。

これらはみんなLotus1-2-3などのソフトによるものだ。

年賀状のお年玉くじの当選番号チェックは，10年以上前にこしらえたBASICのプログラムがまだ活躍している。BEEP命令で，1～4等と5等以下を判別，5等以下の中から5等とハズレをわける。

illustration：Kyoko Takazawa

ワープロとしての利用は多い。年賀状，転勤のあいさつ状，手紙，親族や友人の間での催しごとの通知。あるいは，封書やハガキのあて名，各種のラベルづくり。

通信は実用とホビー面にわかれる。

実用としては，新聞記事の検索や買い物がある。書籍の注文をするとヤマト運輸が宅配してくれる本の宅急便は便利だ。「ヒサゴネット」では事務用品などをあつかっているので，転勤で住所が変わるときには，とりいそぎ住所の印判なども注文する。

あまり実用にいたらないが，高島屋の贈答品ネットも急場には役にたつ。

ホビー面の通信は，なんといっても友人づくりの功績が大きい。これは趣味からビジネスまで境界線がなく，その価値ははかりしれないものがある。

でもこれらのことは，わが家にとって便利なのであって，よその家にそのままあてはまるものではない。誰かのパソコンではなく，自分(家庭)のパソコンを持ったことで，だんだんに効果を見せはじめたわが家流の便利なのだ。

自分のパソコンを，自分のために便利であるように，時間をかけて育てていく。それがほんとうのパーソナルコンピュータをつくることだとすれば，まずはクルマ1台を買うように，パソコンのオーナーにならなくてはいけない。

パソコンがなんの役にたつかと聞かれたら，使ってみて「自分の便利」をさがすしかないと答えたい。

「持ち込み何でも可」の試験

試験監督

寒い寒い大学入試の季節がまたやってきました。悪名高き(本当か?)「共通一次試験第一期生」である僕はというと，相変わらずセンター試験，正確には「大学入学者選抜大学入試センター試験」の監督をしたのでした。教授だろうと何だろうと国立大学の全教官が入試にかかわる仕事(ほとんどは監督)をしたのでしょうから，この試験は何十万人の受験生と合わせて国家的なプロジェクトといって間違いないでしょう。

受験生がご苦労さんであることはいうまでもありませんが，監督のほうも意外としんどいものです。何が起こるか実際のところわからないものですし，もしその対応を誤って，試験場の環境として最も大切といえる「公平さ」を損なうようなことでも起こしたならば，すべてが水の泡というものです。

そういうわけで，今年も監督員一同は，あまり早く試験場に行かないほうが受験生にプレッシャーを与えないだろうとか，直射日光が差し込んでいるからブラインドを少し下ろそうとか，後ろのほうは少し熱気がたまっているから窓を3mmだけ開けようとか，あまりここに立っていたらこの学生が気にするから動こうとか，動くと気にするからやっぱり動くのをやめようかとか，まあそれはもう，実に厳しい状況のなかでくよくよ考えているのです。しかも，うまくいって当たり前ですから，その努力も実は意外にさびしいものです。土曜と日曜をつぶしているのに。

実際，受験会場の物理的環境(雑音，温度，明るさなど)を公平に保つのは大変ですし，さらにたとえば，

1)　ふと外を見ると，ビルの壁の広告の文章のなかに難しい熟語が用いられていた
2)　ふと前を見ると，前の受験生のセーターに英文が縫い込まれていた
3)　街宣カーがスピーカーの音量を上げて何やら難しそうなことを述べたてていた
4)　腕時計に実は計算機能がついていた

などなどによって，一部の人が「しめしめ」となるような状況も推定されるのですが，そこはもちろんきちんと，「英語の入っている上着などはだめ」とか，「計算機能のついている時計はだめ」とか，いちいち規則が設けられているのでした。

公平さの意味

もしも，足し算，掛け算程度の機能を備えた腕時計を受験者が使うことを許すことにしたのならば，確かにそれをつけていたほうが有利でしょうから，それを持っていない受験生との間に不公平を生み出すことになるのでしょう。当然といえば当然の話に思われます。いっぽう，当たり前の話ですが，目の悪い人はメガネをかけたり，コンタクトレンズをつけたりします。これも当然の話です。目が悪ければ，問題用紙さえよく読めませんからね。

でももう一度，計算機能つき電卓の話とくらべてみましょう。こんなふうに考えるのは無理でしょうか?　「視力が悪いからメガネをかける，そうしないと不公平だから」というのと同じように，「足し算や掛け算が苦手だから時計の計算機能を使う，そうしないと不公平だから」。

この論理はあまりに強引だということくらい重々承知していますが，しかし，単に公平かそうでないかという基準だけでは，この問いかけを簡単に却下することはできないような気もします。結局のところ，考えなくてはいけないのは，この試験が受験生のどういう能力について調べようとしているのかということにつきると思います。

そもそも，試験というものはある物差しで受験生に優劣をつけようという大胆不敵な試みですから，その物差しにかかわる部分まで公平にしては，試験そのものの存在意義がなくなるということになるわけです。

単純な計算をする能力をも試そうとしているから，計算機能つきの腕時計はだめなのであり，英単語に関する記憶をも試そうとしているからこそ，英単語のついた上着やセーターを見つけたら，脱がせたり隠させたりしなくてはならないのです。当たり前ですね。

物差しで何を?

いったい何の優劣を判定したいのかということについて，もう少し一般的にいうのならば，それはまあ，高校までの学力がどこまで身についているのかということなのでしょうし，大学で受ける教育に適しているかどうかということなのでしょう。

ところで，もし，体に装着するタイプの電卓の普及がいくところまでいったり，あるいは，ワープロの普及が個人個人のレベルにまでいきわたったのならば，当然，試験の性格もがらりと変わってくるでしょうし，それに応じて試験場への持ち込みに対する制限も変わってくるかもしれません。

僕などが強く思っていることは，時代の情報化の流れに即してというよりは，むしろ情報化の流れを加速するほうに，受験制度，いや教育そのものが変わっていくべきだと考えています。もう少しはっきりいうならば，計算機が容易に取って代わることができるようなことを，無理やり素手の人間にやらせる必要はないし，その能力でもってその人間を判断するようなこともあまり意味がないのではないかというのが，僕の基本的な立場です。

たとえば，漢字の書き方だとか(読み方は当面は重要かもしれませんが)，簡単な計算だとか，単純な丸暗記のようなことだとかは，確かに頭のトレーニング的な意味はあるでしょうが，現在の教育における扱いほどは重要でなくなるべきだと思います。

ちなみに，先日亡くなった安部公房氏も，10年ほど前にすでに，「あれ(書き取りの勉強)の代わりにほかのことをやったらよかったと思う。漢字は機械が書けばいい」と述べていたそうです。

恐ろしい試験

情報化社会が理想的に進展していったときに試験すべきなのは，わざわざ，通常の状態つまりいろいろな情報機器をわざと使えない状態にした，ムキダシの脳なのではなく，日常的な状態における知的能力，いいかえるならば，情報機器や知能機械などを自由に使うことによって到達しうるような，トータルな意味での知的処理能力こそを調べるべきであると僕は思います。

その場合においても，現在の漢字の読み書きや四則演算能力のような基礎的な能力に相当するような初歩的なことは試されなければなりません。それは，そのような情報機器や知能機械の使い方です。

使い方といってもマニュアルに書いてあ

るようなことではなく，どういう場面でどういう機械をどのように用いるべきかというような重要なことがらです。いずれにせよ，このような基礎的な能力も体得していないと，結局のところトータルな知的到達度は上がらないというものです。

しかし，さらに将来において，知能機械なるものがもっと洗練されてくると，そのような使い方的な能力さえ必要なくなります。ユーザーインタフェイスの研究は，機械が真に知的でない場合にのみ成り立つものですし，どのような場合にどのような機器をどのようにというようなマクロなインタフェイスも当然完成されているからです。知能機械が完成すれば，知能機械の存在感さえ逆になくなると思います。

そうしていくと，人々は，機械ができるような比較的単純な知的処理から，1つひとつ解放されていき，しまいには，「何か究極的なこと」しか試験によって測られる必要はなくなってしまうのです。それをひとことでいうのならば，感性とでもいえるのでしょうか？

ただし，感性のほうはすでに何やら研究対象にしている科学者も出てきているようなので，価値観とか，美学とか，感情とかがテストされるのでしょうか？　あな，おそろしや。

思考を支援するツール

ここまでくると，何だか電卓の話からずいぶんと急に知能機械という話に飛躍したように思われるかもしれません。確かに，計算機はちょっとした計算か，せいぜいワープロぐらいしか実際のところできていないようにみえるのもしかたありません。少なくとも，われわれの日常生活の近くにある計算機はそういうものが大多数のようです。

しかしいっぽうで，着実にわれわれの脳味噌に向けて，知能機械の卵(ヒヨコか？)たちがずんずんと攻めてきているということも事実のようです。人間の最も高度な創造的活動である「思考」や「発想」を支援するような試みもすでに行われてきています。

そのような研究例を，参考文献に挙げた折原氏の文章に基づいて紹介してみることにしましょう。このようなものがごく日常的に使われるような世の中はそう遠くないと思います。

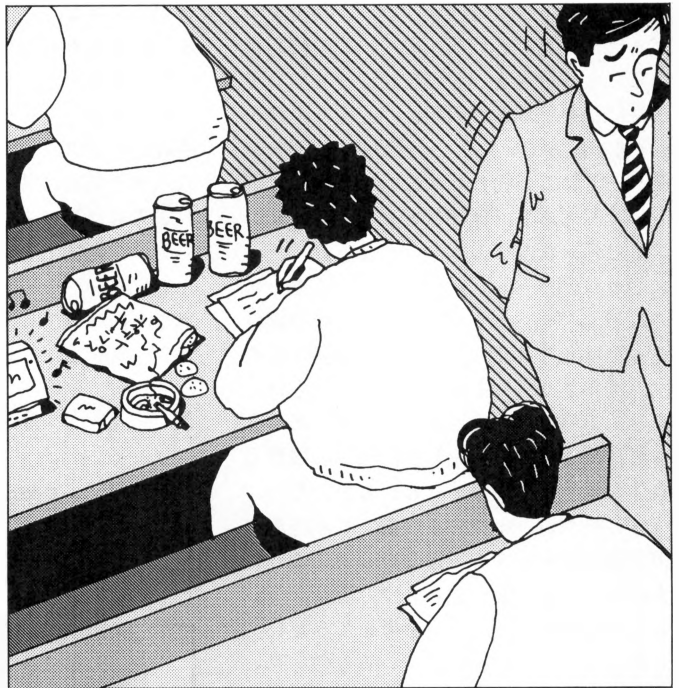
L.F.Youngという人によれば，発想支援のシステムは，秘書レベル，枠組み－パラダイムレベル，生成レベルの3段階に分けることができるそうです。

illustration : Haruhisa Yamada

秘書レベルとしては，ふつうのワープロ，(Macintoshの Hyper Cardのような)ハイパーテキストシステム，データベース，(電子会議室のような)グループウェアなどが含まれます。基本的には，動的な性質を持つ黒板のようなもので，ユーザーはこれを用いることにより，創造的な活動に専念できるようになります。

枠組み－パラダイムレベルとしては，(章立て→筋立て→内容というトップダウンな文書作成をサポートする)アウトラインプロセッサ，(専門家から知識を獲得するための問題解決モデルを持っている)知識獲得支援ツール，さらにAAIというシステムも開発されてきているようです。

このAAIでは，ユーザーは頭に浮かぶいくつかの言葉をシステムに与え，さらに，それらの言葉の間に関係があるかないかを入力します。そうするとシステムは，与えられた情報をもとに，関係のある言葉どうしが近い距離に位置するような言葉の空間配置を計算し，結果をユーザーに知らせるというものです。これにより，ユーザーは空間のなかの空白に位置すべき言葉を考えたりして，発想が支援されるというのです。

3番目の分類である生成レベルに属するとされるシステムもすでに出現しています。すでにIBM PC上に商品化されている有名なシステムが「Idea Fisher」です。このシステムは，「Question Bank」「Idea Bank」「Idea Notepad」から構成されており，ユーザーはまず，QuestionBankと対話することにより，自分の抱えている問題点を明確にし，その次に，その問題点に対する解決案をIdea Bankに入力します。Idea Bankは，入力された語句から連想される語句をデータベースから検索して提示します。このようにしてアイディアをふくらませてから，そのアイディアをまとめるためのエディタがIdea Notepadなのだそうです。

ほかにもいくつかの生成レベルに属するシステムが紹介されていますが，興味深いのは，渡辺勇氏による「Keyword Associator」です。それは，電子ニュースの記事をもとに連想辞書を自動生成するというものです。連想辞書そのものも計算機が自由に処理できるネットワーク上のテキスト列から読み込むというのは，安直にみえるかもしれませんが極めて実用的な手法だといえましょう。

このようなシステムが次々と進化して，多くの低レベルの処理から人間を解放した世の中において，われわれは受験生と呼ぶ人々の頭のなかの何をどのようにテストするのでしょうか？　そろそろ具体的に考えないと間に合わないでしょうね。

参考文献

折原良平，“発想支援システムの動向”，「情報処理」1993年1月号，81-87pp.

X-OVER・NIGHT
（クロスオーバーナイト）
［第32話］変わってきた

TAKAHARA HIDEKI 高原 秀己

景気低迷といわれて久しいが，個人消費の落ち込みは高度成長期以降，最大級だという。たとえば昨年のデパートの全国販売総額は1991年よりも3.3％減っており，戦後混乱期の不明な時期を除けば，初めてのマイナス成長だという。とくにこの傾向は昨年でも月を追うごとにひどくなっていて，12月だけをとると，前年よりも売り上げが8.1％も落ちている。これは11人に1人がデパートに買い物に行かなくなったのに等しい状況であり，ほぼひとまわりシュリンクした購買状況であるといえよう。

べつにデパートで買い物しなくてもいいのだが，スーパーや専門店チェーンの統計でも，昨年1年間で0.5％しか伸びておらず，例年よりずっと低い。ということは，単なるデパート離れにとどまらず，あちこちで指摘されているとおり，個人の購買力自体が低下しているといわざるをえない。

といいつつ，べつにこの欄で日本経済の消費全般を考察する気はない。逆にこういう時期であるにもかかわらず，流行しているものは何か，という点を考えてみたい。

身近なところで流行しているものを思い浮かべると，カラオケボックスと超大型ゲームセンターの2つが挙げられるだろう。

カラオケボックスの増え方はすさまじいのひとことにつきる。新宿，渋谷，池袋といった都心部に限らず，郊外地，地方都市にも続々と増えている。実際に行きつけの飲食店があっという間にカラオケボックスに化けてしまっているケースが実に多いし，環状7号線とか8号線とかを走っていても，怪しげなホテル(こちらは行きつけではない，念のため)が突然カラオケボックスに変身してしまっているのもよく見る。

どうしてこんなにカラオケボックスが必要なのだろうか，と思ってしまうのだが，実際に行ってみると，よくもまあこれだけ人がいるものだと感心してしまうほどの賑わいだ。先日など「金曜はさすがに混んでいるだろうから，土曜にしよう」といって，土曜の深夜に予約なしで行ってみると，あわや待たされかねない状態だった。

カラオケボックスのいいところは，メンバーだけで1台の機械を独占できるうえ，そこそこの飲み食いもできることだろう(そこそこと書いたが，店によってはレミーマルタンあたりの洋酒まで用意してある)。もちろんカラオケ自体はスナックやパブでもできるのだが，ほかの客と交替になるうえ，とんでもなく古い演歌や軍歌を壊滅的音程で聴かされるのに耐えなくてはならない。これもちょっとつらいところ。カラオケボックスなら，ちょっとしたパーティールーム感覚もあるし，長所は多い。

いっぽうの超大型ゲームセンター。二子玉川にできた「ナムコワンダーエッグ」や渋谷の「ドクタージーカンズ」をはしりに，六本木や新宿に続々とできている。

こちらは先日まで行ったことがなかったのだが，実際に行ってみると入り口にはちゃんとUFOキャッチャーがひしめいている。奥や上のフロアに行くと，コインゲームがズラズラと並んでいる。ただしこれだけではなく，別のフロアに特別室よろしくカジノ風コーナーを設けてあるという仕掛け。それほど斬新というイメージはないのだが，カジノ風コーナーがあることで特殊なテイストがある点がミソのようだ。実際にうなるほど客が入っていたのだから，人気は上々だ。商店街のゲームセンター特有のどこか暗い雰囲気がないのもいい。

さて，ここで客の立場でカラオケボックスと超大型ゲーセンの双方の共通点を考えてみたいのだが，顧客は20歳代のサラリーマンを中心に分布している。ゲームセンターだからといって学生ばかりがたむろしているわけでは決してない。なにしろ18歳未満は入場お断りの店まである。で，このゾーンの人が比較的安価に深夜まで遊べるプレイスポットとなっている点が指摘できる。

一般的に，ショットバーで一杯というのを除けば，居酒屋クラスで3,000円前後，スナックやパブラウンジで5,000円前後，クラブなど高級な店で飲むともう1万円を超えてしまうのが現在の相場だろう。こういう店は飲み食い以外では，これといった楽しみはない。

その点，カラオケボックスにせよ，超大型ゲーセンにせよ，かなり長い時間遊ぶとしても5,000円未満でなんとかなる。さらに，ここで遊ぶ人はだいたいがちゃんと電車で帰宅しているから，出費総額まで考えると，かなり割安ということになる。

もうひとつは，夜の娯楽の多様化を求める欲求が花開いたともいえる。この価格帯のプレイスポットといえば，ディスコが代表格だったのだろうが，「ジュリアナ東京」の人気とは裏腹に，老舗ディスコの閉店などもあってか，ディスコ愛好者は激減しているようだ。なにしろ仕事が終わってから行くのに，やれ「その服装ではご遠慮願います」だの「男性のみはお断り」だのいわれるのだから，よほどいいことでもないと自然に足は遠のく。でもって「よほどいいこと」なんぞ，そうそうはないのである！

かたや，夜の銀座に閑古鳥が鳴いている状態はまだ好転していない。なにしろすぐにタクシーが拾えるのだから，推して知るべしである。同様に六本木や新宿の酒場も，そこそこ高い店はいぜんガラガラ。価値観の変遷によって，ただ単に飲むだけの店に飽きがきていたこともあるのだろう。

このように，バブル崩壊に伴う節約基調と新しい遊びへのニーズがうまく合体して生まれたカラオケボックスと超大型ゲーセンの例をみると，世の中でいわれている「変化の時代」というのもうなずける気がする。デパートに置いてある商品の内容も徐々に変わっているし，スポーツや旅行の楽しみ方も変化している。

もちろんそれと同様に，パソコンを取り巻く環境も徐々に変わってきたことはいうまでもない。ラップトップの普及で気軽に持ち運びできるようになったし，DOS/VブームでI実際はVTR程度の，さほど高くない品物だったことが暴露された。いよいよ次に変わるのが「使われ方」であれば，とても楽しみなのだが。

illustration : Haruhisa Yamada

〈対応機種一覧〉●MZ-80K/C/700/1500●MZ-80B/2000●MZ-2500/2861●X1●X1turbo/Z●PC-8001/8801/88●SMC-777/C●PASOPIA/5●PASOPIA 7●FM-7/77/AV●PC-286/386/486/9801/98/9821●X68000
掲載されたプログラムの利用には各機種用のS-OS"SWORD"システムが必要です。

第130部　シューティングゲームコアシステム作成法(1)

●シューティングゲーム

今月から，シューティングゲームを作るための支援システムを制作するシューティングゲームコアシステム作成法の連載が始まりました。

連載の趣旨としては，ゲームシステムの制作方法をより具体的に紹介していこう，というものです。

今回は，連載初回ということで，画面制御関係サブルーチンの試作を行っています。背景とキャラクター表示のための画面として，合計3画面用意するという，結構，豪華なものです。すでに，ELFES IVでは実現しているので不可能ということはありません。当時はかなりの衝撃を与えましたが，アルゴリズムを知ってしまえば意外に簡単だと思うでしょう。まったく同じというわけではありませんが，今回の解説でどのような手法を使って実現しているか理解してください。

しかし，こういったシステムの宿命として，使う側の賛同を得られなければ自然消滅してしまうことがあります。複雑になればなるほど，作者本人にしかわからないものですし，かといってハンパなものでは見向きもされません。このあたりのバランスが非常に難しいものです。

ELFESシリーズに触発されて，このプログラムの制作を始めた坂巻氏ですが，はたしてELFESシリーズを超えるものを作り上げることができるのでしょうか。

また，この連載で作成する予定のものは，とりあえずシューティングゲームコアシステムですが，使い方によっては，現在あるスプライトゲームならたいていのものを作れる汎用性のあるシステムにしたい，ともいっています。

風呂敷を広げるだけ広げているような感じもしますが，ちょっと注目してみたいですね。

●プログラミング

リアルタイムゲームに求められる速度をS-OSの世界で実現するには，結構難しいものがあります。

ゲームを表現する手段がキャラクタのみに限られるのはいいとして，そのキャラクタ表示方法がS-OSに依存しているのが，問題です。

できる範囲にやりたいことを納めるのもひとつのテクニックですが，やっぱり目標は高く持ちたいもの。すると必然的に泥沼の高速化，アルゴリズムの徹底的な見直しを行わなくてはなりません。

プログラムをコーディングするときに行う最適化はもちろんですが，いかにして最小の手間で実現するかを考えるのも，かなり根気がいる作業でしかも地道かつ泥臭いところがありました。

しかし，その高速化を突き詰めたアルゴリズムとコーディングされたプログラムは，シンプルで美しいものです。また，それによって得られるものは大きく，プログラマにとってはかけがえのない経験となります。さらに得られた作品を活用すれば，それをもとに新しい作品を作り上げていくこともできます。

何かを作り上げることを目標に突き進む，自分のやりたいことをやる。コンピュータによってそれができるのなら，非常に楽しいことだと思いませんか。

●S-OSの系譜（42）

1990年4月号では，ファジィコンピュータシミュレーション「I-MY」が発表されました。ファジィという言葉は，電子炊飯器や電気洗濯機などに搭載され始めた頃話題を呼んだ，ファジィ理論として読者の皆さんにも多少馴染みがあることでしょう。

ファジィの言葉の意味するところは，曖昧というものです。簡単にいえば，0/1というデジタルなものから離れ，「だいたい」とか「そのくらい」といった表現の意味をもたせることができたのです。

この「I-MY」は，1と0の2つの値しかもたないコンピュータの世界へ，0.5という中間値を持ち込んだ3値論理の推論エンジンを使い，ファジィコンピュータをシミュレートしていました。ファジィ理論を理解したうえで使わなくてはならない，という制約もありましたが，逆にわかって使えばかなり面白い体験をすることができたものです。

S-OSの世界では，かなり異色なこのプログラム。いま一度思い返してみることによって，新しいものが見えてくるかもしれませんね。

1993■インデックス

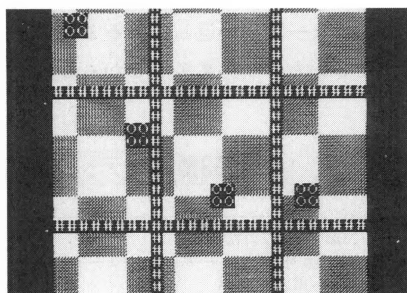
THE SENTINEL

全機種共通
S-OS"SWORD"要

シューティングゲーム

コアシステム作成法(1)

Sakamaki Katsumi
坂巻 克巳

今月からシューティングゲームを制作することを目的とした，コアシステムの制作を行っていきます。今回は大まかな仕様の決定と，画面制御関係のルーチンを作ります。

コアシステムの制作

現在まで，S-OSにはいろいろなプログラムが発表されてきました。開発言語やプログラム支援ツール，ゲームなどなど。

かなり制約のある世界ながら，どれも個性的で味のあるものが発表されています。また，システムに限らずゲームでもキャラクタグラフィックだけで多彩な表現がされていましたね。

そのなかでも，特に印象に残ったものは，REDA，SLANG，そしてELFESシリーズでした。アセンブラ人間の僕にとって，REDAのシンプルさ使いやすさはプログラム開発をするうえで手放せない存在ですし，そのアセンブラ人間の僕に高級言語の面白さを教えてくれたSLANGも，ちょくちょく使っています。ELFESシリーズにいたっては，新作が発表されるたびに新しいことをやって，S-OSでのゲームはここまでできる，ということを知らしめてくれました。

今回，制作しようと目論んでいるのは，ELFES IVを意識したシューティングゲームのためのコアシステムです。シューティングゲームを制作するうえで，役に立つサービスコールをまとめたものを目指していきます。

こういったゲームパッケージといえば，以前にもBEMSというゲーム制作のためのパッケージがありました。簡単に概念を復習すると，BEMSはゲームの要素を背景(Back)，敵(Enemy)，ミサイル(Missile)，自機(Ship)に分け，ゲームに必要で面倒な座標管理，衝突判定を自動的に行ってくれたのです。せっかくいい見本があるのですから，BEMSを参考にしつつ，さらに多機能，高機能なコアシステムを作りたいと考えています。こういったものを連載という形で制作するのは初めてですが，がんばりますのでよろしくお願いします。

仕様を考える

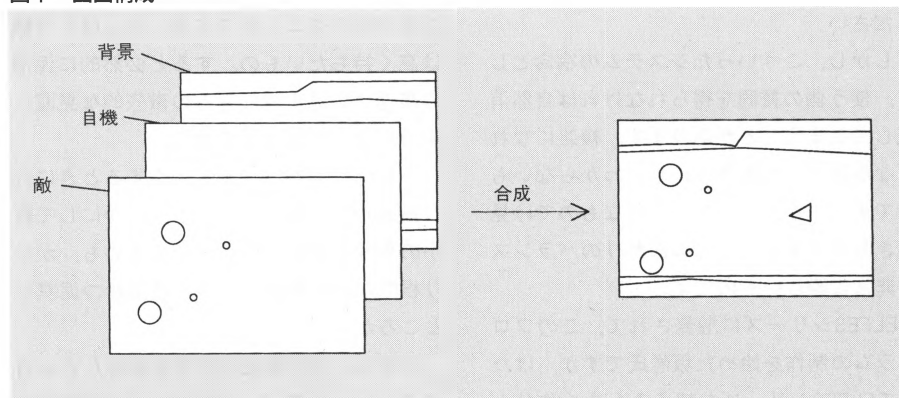
まずは，キャラクターを描画するための画面制御関係の仕様を決定します。

作りたいものはシューティングゲームを支援するシステムです。おおまかに考えて，シューティングゲームに登場するキャラクターは，

1)　自機キャラクター
2)　敵キャラクター
3)　背景

以上の3種類で構成されていることでしょう(自機の弾，敵弾も含む)。まずは，自分が操作するキャラクター，コンピュータによって操作される敵キャラクター，障害物としての役割をはたす背景，という具合です(このあたりの概念はBEMSとほぼ同じ)。ゲームでは，種類ごとに表示画面を用意して，最終的にひとつの表示画面に合成するのが理想といえます(図1)。

また，このように種類別に画面を用意することで，キャラクターどうしの当たり判定が多少有利になります。シューティングゲームでの当たり判定は，1)自機キャラクターと2)敵キャラクター，3)背景に対して行われています(敵キャラクターと背景は考えない)。表示画面が分けられていないと，いちいちキャラクター単位で座標比較をして判定を行わなければなりませんが，きちんと種類ごとに分けられていれば，自機キャラクターの位置に，敵キャラクター，背景の表示画面に何が書き込まれているか

図1　画面構成

を確認するだけですむからです(図2)。つまり，空白以外のキャラクタが書き込まれていれば，衝突判定が起きたと考えられます。

といっても，S-OSにはごく簡単なキャラクタ表示ルーチンしかありません。サービスコールがない＝実現不可能と考え，素直にあきらめてもいいのですが，それでは悲しすぎますね。要するに，S-OSがやってくれないなら，自分でこれらの処理ルーチンを作ればいいだけだ，と思い込み，どのような手法によって実現可能となるか探っていきます。

といっても，うだうだと考えていきつくところは，仮想画面の概念を持ち出すことぐらい。ほかにもないかな……と考えてみましたが，僕の経験からいってもこれ以上有効な手段は見つからないし，かなり応用範囲の広いものですから問題はないでしょう。

図2　当たり判定

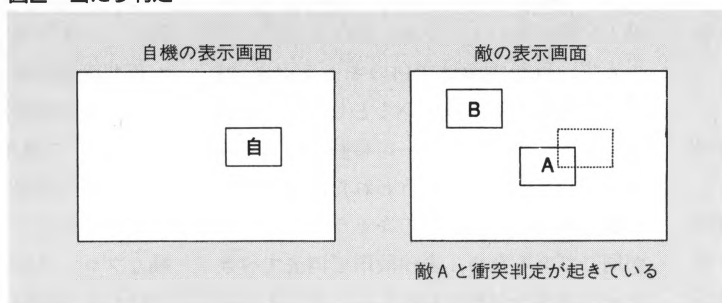



図3　仮想画面構成

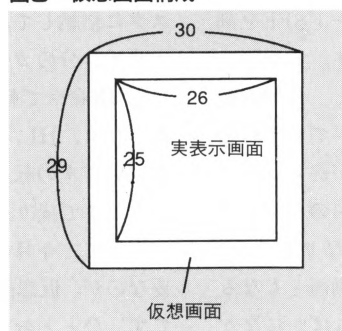
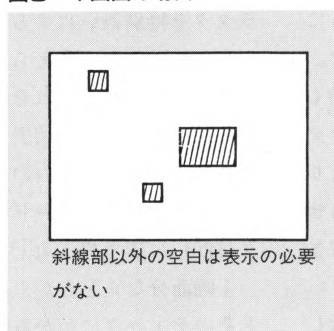
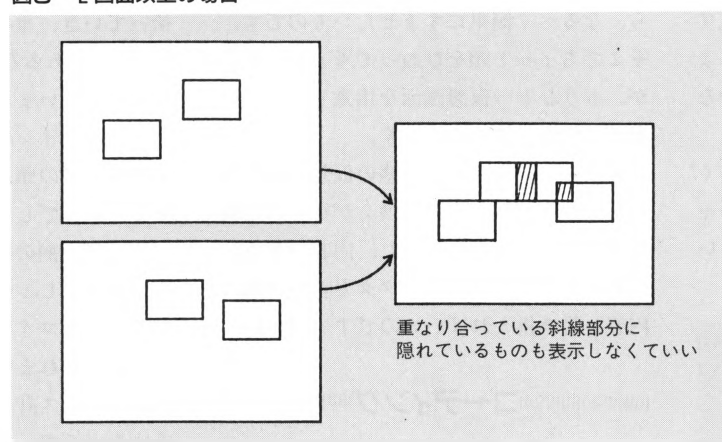



図5　1画面の場合

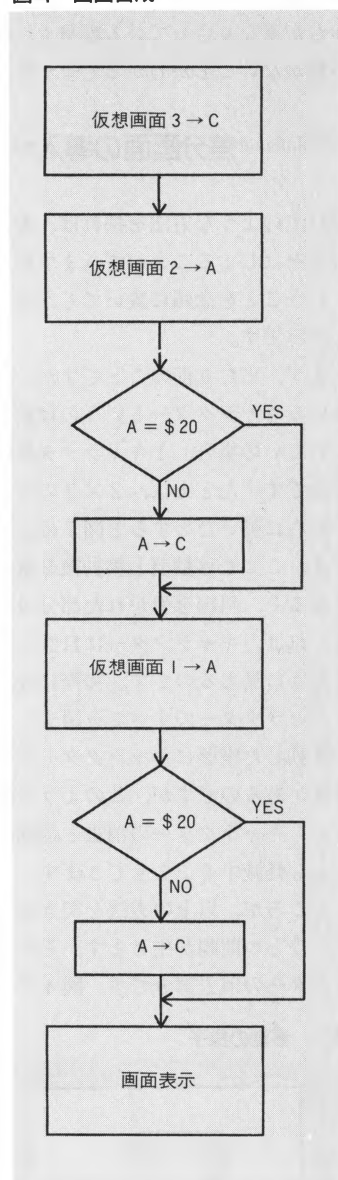
図6　2画面以上の場合

図4　画面合成

ここまでで，用意すべきものは3種類の仮想画面，そしてそれらの画面を合成する処理ルーチンということになります。なお，仮想画面のサイズは30×29キャラクタ分としておき，実表示画面は26×25キャラクタ分とします(図3)。肝心のゲームシステムは，とりあえず縦スクロールシューティングとだけ考えておきましょう。

仮想画面の合成方法

結論から先にいってしまうと，仮想画面を合成するためには，図4のようなアルゴリズムで処理することになります。優先順位の低い仮想画面から内容を順次拾っていって，優先順位の高い仮想画面が空白なら低い仮想画面にあったキャラクタを通し，そうでなかったらキャラクタの入れ替えを行って，最後に残ったキャラクタを画面に表示してやるのです。

仮想画面というのは，あくまでもユーザーが勝手にメモリ上に用意した画面であって，そこに何かを書き込んだからといって画面上にはなんの変化も起きません。必ず，書き込んだ仮想画面の内容に従って，表示画面にキャラクタを表示してやらなければならないのです。そのためこのような仮想画面の合成を行い，その結果をディスプレイに反映していくことになります。

しかし，仮想画面を用意して画面合成がうまくいったとしても，馬鹿正直に仮想画面に書き込まれているキャラクタを表示していったのでは，とても処理しきれないのは明白です。かといって，機種別にプログラムを用意して，直接I/Oをいじることで実現しても意味がありません。どうしても姑息な高速化を考える必要があります。

問題なのは表示キャラクタ個数ですから，その表示するキャラクタをいかにして減らすことができるのかを考えましょう。

最初に仮想画面が1画面あった場合（図5）ですが，見てのとおりキャラクタが表示されている部分以外は，空白キャラクタが置かれることになります。つまり，仮想画面をスキャンしていって，空白以外のキャラクタが見つかったときのみ，キャラクタを表示すればいいのです。

次に仮想画面が2画面以上あった場合(図6)では，1画面のときと同様に空白のキャラクタの表示をキャンセルできます。

そして，仮想画面に置かれたキャラクタどうしが重なり合っている部分も，表示する必要がないことがわかるでしょう。

差分画面の導入

以上のような方法を採れば，ある程度の表示をはしょることができますが，ゲームということを念頭に置いてもう少し突っ込んでみます。

まず，当たり前のことですが，ゲームにおけるキャラクターというのは動きます。たいていの場合，1キャラクタ単位で動くものです。たとえば，2×2のキャラクターが右に動いたとすると図7のようになります。ここで移動前と移動後を重ね合わせてみると，斜線を引かれた部分のみを書き換えれば，キャラクターは自然に動いているように見えるのです。実際には，表示したキャラクターのすべてを消去して，新たに移動した座標にキャラクターを書き直す必要があるのですが，このような操作によって，キャラクターの消去と描画を多少なりとも軽減することもできます。

ところが，以上の方法を突き詰めていくと，ひとつ問題が生じます。それは，キャラクターの消去部分です。図4でうまくいくような話しぶりでしたが，実際にはある細工を施さないと，正常に動きません。

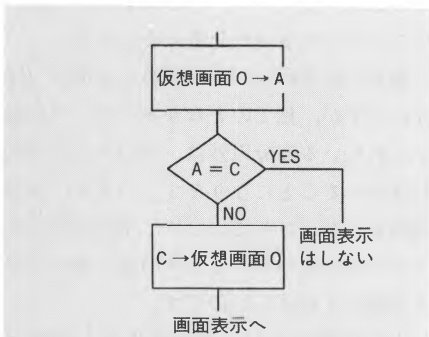



図7 移動の様子

図8 完全な画面合成

図4の画面表示の前に以上の処理を追加

まず，仮想画面は空白のキャラクタ（$20）で埋めつくされているとし，そこで前述のようにキャラクターの移動を行ってみます。すると，消去が行われた部分とほかの仮想画面の空白部分のキャラクタコードが同じであるため，表示段階で消去すべきキャラクターがわからなくなってしまうのです。対処方法としては，消去されたキャラクタを特別扱いにする（コード$FFを割り当てる）ことも考えられます。しかし，複数の仮想画面でそれを行うと，仮想画面ごとにいちいちその消去コードで判定が必要になりますし，空白の表示が終わったあとには，その消去コードを空白のキャラクタ($20）に置き換えなければなりません。

1画面分ならともかく，3画面ともなるとそのチェックにかかる時間もばかになりません。結局消去するのは1度なんですから，なるべく簡単にすませたいものです。そこでちょっと頭をひねって考えついたのが，もうひとつ仮想画面を用意して，それを差分画面とする方法です。

要するに合成された状態の仮想画面を保存しておき，新しく書き換えが行われた仮想画面と差分画面を比較し，内容の変更が行われていたら，キャラクタと差分画面の内容を書き換えればいいのです（図8）。

コーディング

それでは，作るべきもののアルゴリズムがある程度固まったところで，コーディングに移ります。

今月は，とりあえず仮想画面合成のサンプルプログラム的なものを作ってみました（リスト1）。内容は，チェック模様の背景がスクロールし，“#”で書かれた格子と背景の間をキャラクタが動き回るというだけのものです。サンプルのわりにはちょっと長めのリストですが，実際どれぐらいの速度で動いているか確認するにはいいでしょう。

このリスト1の中で理解してもらいたいのが，

・仮想画面の初期化（VRAMINIT）
・背景のスクロール（BGSCROLL）
・仮想画面の合成（PAGEMIX）

以上，3つのサブルーチン，そして仮想画面ワークエリア（VRAM0〜3）です。

仮想画面の初期化(VRAMINIT)については，仮想画面0〜3をスペースキャラクタ（$20）で埋めつくしているだけですので，説明の必要はないと思います。

次に背景のスクロールですが，これも単純なブロック転送を下から順に行っているだけです。最初に最下段のアドレスをDEレジスタに，最下段−1のアドレスをHLレジスタに格納して，BCレジスタに格納された1ライン分のカウンタ（30バイト）を一気にLDIR命令で転送しています。転送が終わったら，HL，DEレジスタから60バイトを引いて次の転送に備え，転送元が最上段になるまで繰り返しています。

そして，今月のリストの中でいちばん重要なのが，仮想画面を合成するPAGEMIXです。ひととおりリストを眺めると，優先順位の低い背景から順に仮想画面の内容を拾っていき，重なり合う仮想画面の内容が$20以外であるなら，キャラクタの入れ替えを行っています。1ライン分が終わったら，次のラインの先頭アドレスを計算して，またループの頭に戻っている，というのが見てとれるでしょう。

ここで，勘のいい人は疑問に思ったところがあるでしょう。すでにわかってしまった人は，セコイことをしているなあ，と思ったかもしれません。それは，各仮想画面間のアドレス計算を上位バイトのみで行っている点です。

わけがわからない人は，仮想画面のワークエリアであるVRAM0〜3で，ワークを1024バイトを確保している部分に注目してください。仮想画面は，30×29キャラクタ分を用意するだけでいいのですから，870バイト確保するだけで用が足ります。これは，使われない154バイトを犠牲にしても，処理速度を稼ぎたかったからなのです。

Z80のアドレスは16ビットで表され，計算も16ビット単位で行うのが普通です。しかし，リストを見てもわかるとおり，余っているレジスタはないし，かといって裏レジスタを総動員するとこんがらかるし（カウンタに使ってますけど)，スタック関係の命令は重くて使いたくない。いちばん簡単でてっとり早い方法が，ワークを256バイト単位で区切り，アドレス計算を単純にする

ことだったのです。

まあ，余ったメモリについては再利用をするかどうかわかりませんが，しばらくはっておくことにします。

完成させます

解説を聞いている限りでは，結構複雑なことをしなければならないような気がしますが，実際にコーディングされたリストを見ると，案外スッキリしたものに仕上がっているのがわかると思います。

確かに，概念としては使い古された常套手段といえるものです。しかし，実際にやってみようとすると処理速度の問題が……と躊躇してしまう場合がありませんか？ 少なくとも僕はそうでした。はっきりいって，実物をこの手で作って動かしてみるまで，不安があったのです。やってみたら意外にうまくいってしまった，そんな感じです。冒頭で述べたように，やはりいいものを作りたいですからね。やるだけやってみなくちゃ。

来月は，これらの画面制御関係サブルーチンに，キャラクター表示ルーチンを加えた，よりシューティングゲームシステムらしいものを完成させる予定です。お楽しみに。

リスト1

```
0000                 1
0000                 2  ;*********************************
0000                 3  ; SHOOTING  GAME
0000                 4  ;             CORE SYSTEM
0000                 5  ;*********************************
0000                 6
9000                 7            ORG     $9000
9000                 8
1FF4 P               9  #PRINT  EQU      $1FF4
201E P              10  #LOC    EQU      $201E
1FC4 P              11  #BELL   EQU      $1FC4
1FBE P              12  #PRTHL  EQU      $1FBE
1FC1 P              13  #PRTHX  EQU      $1FC1
1FEE P              14  #LTNL   EQU      $1FEE
1FD0 P              15  #GETKEY EQU      $1FD0
9000                16
9000                17  TEST:
9000 3E 0C          18       LD     A,$0C
9002 CD F4 1F       19       CALL   #PRINT
9005 CD 78 91       20       CALL   VRAMINIT
9008 CD B5 90       21       CALL   PAGE1SET
900B                22  TEST2:
900B CD 1F 90       23       CALL   PAGE2CHR
900E CD EC 90       24       CALL   BGTEST
9011 CD 18 91       25       CALL   BGSCROLL
9014 CD 33 91       26       CALL   PAGEMIX
9017 CD D0 1F       27       CALL   #GETKEY
901A FE 20          28       CP     $20
901C 20 ED          29       JR     NZ,TEST2
901E C9             30       RET
901F                31
901F                32  ; PAGE2 CHR PUT
901F                33
901F                34  PAGE2CHR:
901F DD 21 C7 91    35       LD     IX,CHRWORK
9023 06 04          36       LD     B,04
9025                37  PGM2:
9025 C5             38       PUSH   BC
9026 CD 32 90       39       CALL   CHRMAIN
9029 C1             40       POP    BC
902A 11 04 00       41       LD     DE,4
902D DD 19          42       ADD    IX,DE
902F 10 F4          43       DJNZ   PGM2
9031 C9             44       RET
9032                45
9032                46  CHRMAIN:
9032 CD 89 90       47       CALL   ADDRCAL
9035 3E 20          48       LD     A,' '
9037 CD 9D 90       49       CALL   CHRPUT
903A                50
903A DD 7E 00       51       LD     A,(IX+0)
903D FE 02          52       CP     2
903F 38 04          53       JR     C,DXREV
9041 FE 1C          54       CP     28
9043 38 10          55       JR     C,MOVEX
9045                56  DXREV:
9045 47             57       LD     B,A
9046 DD 7E 02       58       LD     A,(IX+2)
9049 EE FF          59       XOR    $FF
904B 3C             60       INC    A
904C DD 77 02       61       LD     (IX+2),A
904F 80             62       ADD    A,B
9050 DD 77 00       63       LD     (IX+0),A
9053 18 08          64       JR     YCHK
9055                65  MOVEX:
9055 47             66       LD     B,A
9056 DD 7E 02       67       LD     A,(IX+2)
9059 80             68       ADD    A,B
905A DD 77 00       69       LD     (IX+0),A
905D                70  YCHK:
905D DD 7E 01       71       LD     A,(IX+1)
9060 FE 02          72       CP     2
9062 38 04          73       JR     C,DYREV
9064 FE 1B          74       CP     27
9066 38 10          75       JR     C,MOVEY
9068                76  DYREV:
9068 47             77       LD     B,A
9069 DD 7E 03       78       LD     A,(IX+3)
906C EE FF          79       XOR    $FF
906E 3C             80       INC    A
906F DD 77 03       81       LD     (IX+3),A
9072 80             82       ADD    A,B
9073 DD 77 01       83       LD     (IX+1),A
9076 18 08          84       JR     CPUT
9078                85  MOVEY:
9078 47             86       LD     B,A
9079 DD 7E 03       87       LD     A,(IX+3)
907C 80             88       ADD    A,B
907D DD 77 01       89       LD     (IX+1),A
9080                90  CPUT:
9080 CD 89 90       91       CALL   ADDRCAL
9083 3E 4F          92       LD     A,'O'
9085 CD 9D 90       93       CALL   CHRPUT
9088 C9             94       RET
9089                95
9089                96  ; VRAM PUT ADDR
9089                97
9089                98  ADDRCAL:
9089 DD 7E 01       99       LD     A,(IX+1)        ;Y*30
908C 11 1E 00      100       LD     DE,30
908F CD A8 90      101       CALL   MUL8
9092 11 00 B4      102       LD     DE,VRAM2
9095 19            103       ADD    HL,DE
9096 DD 4E 00      104       LD     C,(IX+0)        ;ADDR+X
9099 06 00         105       LD     B,00
909B 09            106       ADD    HL,BC
909C C9            107       RET
909D               108
909D               109  ; 2*2 CHARACTER PUT
909D               110
909D               111  CHRPUT:
909D 11 1D 00      112       LD     DE,29
90A0               113  PGC2:
90A0 77            114       LD     (HL),A
90A1 23            115       INC    HL
90A2 77            116       LD     (HL),A
90A3 19            117       ADD    HL,DE
90A4 77            118       LD     (HL),A
90A5 23            119       INC    HL
90A6 77            120       LD     (HL),A
90A7 C9            121       RET
90A8               122
90A8               123  ; 16bit*8bit
90A8               124
90A8               125  MUL8:
90A8 21 00 00      126       LD     HL,0000
90AB 06 08         127       LD     B,08
90AD               128  ML83:
90AD 29            129       ADD    HL,HL
90AE 87            130       ADD    A,A
90AF 30 01         131       JR     NC,ML82
90B1 19            132       ADD    HL,DE
90B2               133  ML82:
90B2 10 F9         134       DJNZ   ML83
90B4 C9            135       RET
90B5               136
90B5               137  ; PAGE1 # PUT
90B5               138
90B5               139  PAGE1SET:
90B5 21 0E B9      140       LD     HL,VRAM1+270
90B8 11 0F B9      141       LD     DE,VRAM1+271
90BB 01 1D 00      142       LD     BC,29
90BE 3E 23         143       LD     A,'#'
90C0 77            144       LD     (HL),A
90C1 ED B0         145       LDIR
90C3               146
90C3 21 58 BA      147       LD     HL,VRAM1+600
90C6 11 59 BA      148       LD     DE,VRAM1+601
90C9 01 1D 00      149       LD     BC,29
90CC 77            150       LD     (HL),A
90CD ED B0         151       LDIR
90CF               152
90CF 1E 0A         153       LD     E,10
90D1 CD DA 90      154       CALL   TATELINE
90D4 1E 14         155       LD     E,20
90D6 CD DA 90      156       CALL   TATELINE
90D9 C9            157       RET
90DA               158
90DA               159  TATELINE:
90DA 21 00 B8      160       LD     HL,VRAM1
90DD 16 00         161       LD     D,00
90DF 19            162       ADD    HL,DE
90E0 11 1E 00      163       LD     DE,30
90E3 06 1D         164       LD     B,29
90E5 3E 23         165       LD     A,'#'
90E7               166  TTL2:
90E7 77            167       LD     (HL),A
90E8 19            168       ADD    HL,DE
90E9 10 FC         169       DJNZ   TTL2
90EB C9            170       RET
90EC               171
90EC               172  BGTEST:
90EC 3A 87 91      173       LD     A,(BGCNT)
90EF 3D            174       DEC    A
```

```
90F0 20 17          175        JR      NZ,BGT2
90F2 21 8B 91       176        LD      HL,BGDATA1
90F5 3A 8A 91       177        LD      A,(BGFLAG)
90F8 FE 00          178        CP      0
90FA 28 03          179        JR      Z,BGT3
90FC 21 A9 91       180        LD      HL,BGDATA2
90FF                181  BGT3:
90FF EE 01          182        XOR     1
9101 32 8A 91       183        LD      (BGFLAG),A
9104 22 88 91       184        LD      (BGADDR),HL
9107 3E 05          185        LD      A,05
9109                186  BGT2:
9109 32 87 91       187        LD      (BGCNT),A
910C 2A 88 91       188        LD      HL,(BGADDR)
910F 11 00 B0       189        LD      DE,VRAM3
9112 01 1E 00       190        LD      BC,30
9115 ED B0          191        LDIR
9117 C9             192        RET
9118                193
9118                194  BGSCROLL:
9118 21 2A B3       195        LD      HL,27*30+VRAM3
911B 11 48 B3       196        LD      DE,28*30+VRAM3
911E 3E 1C          197        LD      A,28
9120                198  BGS2:
9120 01 1E 00       199        LD      BC,30
9123 ED B0          200        LDIR
9125 01 3C 00       201        LD      BC,60
9128 B7             202        OR      A
9129 ED 42          203        SBC     HL,BC
912B EB             204        EX      DE,HL
912C ED 42          205        SBC     HL,BC
912E EB             206        EX      DE,HL
912F 3D             207        DEC     A
9130 20 EE          208        JR      NZ,BGS2
9132 C9             209        RET
9133                210
9133                211  PAGEMIX:
9133 16 00          212        LD      D,00         ;DISP START
9135 21 3E B0       213        LD      HL,VRAM3+62
9138                214
9138 D9             215        EXX
9139 06 19          216        LD      B,25
913B                217  PM1:
913B D9             218        EXX
913C 1E 05          219        LD      E,05
913E                220
913E 06 1A          221        LD      B,26
9140                222  PM2:
9140 4E             223        LD      C,(HL)          ;PAGE3 CHR GET
9141                224
9141 24             225        INC     H               ;NEXT PAGE
9142 24             226        INC     H
9143 24             227        INC     H
9144 24             228        INC     H
9145                229
9145 7E             230        LD      A,(HL)          ;PAGE2 CHR GET
9146 FE 20          231        CP      $20
9148 28 01          232        JR      Z,PM3
914A 4F             233        LD      C,A
914B                234  PM3:
914B 24             235        INC     H               ;NEXT PAGE
914C 24             236        INC     H
914D 24             237        INC     H
914E 24             238        INC     H
914F                239
914F 7E             240        LD      A,(HL)          ;PAGE1 CHR GET
9150 FE 20          241        CP      $20
9152 28 01          242        JR      Z,PM4
9154 4F             243        LD      C,A
9155                244  PM4:
9155 24             245        INC     H               ;NEXT PAGE
9156 24             246        INC     H
9157 24             247        INC     H
9158 24             248        INC     H
9159                249
9159 7E             250        LD      A,(HL)          ;PAGE0 CHR GET
915A B9             251        CP      C
915B 28 0A          252        JR      Z,PM6
915D                253
915D 71             254        LD      (HL),C          ;CHR REWRITE
915E EB             255        EX      DE,HL
915F CD 1E 20       256        CALL    #LOC
9162 EB             257        EX      DE,HL
9163 79             258        LD      A,C
9164 CD F4 1F       259        CALL    #PRINT
9167                260  PM6:
9167 7C             261        LD      A,H
9168 D6 0C          262        SUB     12
916A 67             263        LD      H,A
916B                264
916B 23             265        INC     HL
916C 1C             266        INC     E
916D                267
916D 10 D1          268        DJNZ    PM2
916F                269
916F 23             270        INC     HL
9170 23             271        INC     HL
9171 23             272        INC     HL
9172 23             273        INC     HL
9173 14             274        INC     D
9174                275
9174 D9             276        EXX
9175 10 C4          277        DJNZ    PM1
9177                278
9177 C9             279        RET
9178                280
9178                281  VRAMINIT:
9178 21 00 B0       282        LD      HL,VRAM3
917B 11 01 B0       283        LD      DE,VRAM3+1
917E 3E 20          284        LD      A,' '
9180 77             285        LD      (HL),A
9181 01 00 10       286        LD      BC,1024*4
9184 ED B0          287        LDIR
9186 C9             288        RET
9187                289
9187                290  ; WORK AREA
9187                291
9187                292  BGCNT:
9187 05             293        DB      05
9188                294  BGADDR:
9188 8B 91          295        DW      BGDATA1
918A                296  BGFLAG:
918A 00             297        DB      00
918B                298
918B                299  ; BG TEST DATA
918B                300
918B                301  BGDATA1:
918B 7B 7B 7B 7B    302        DB      $7B,$7B,$7B,$7B,$7D,$7D,$7D,$7D
918F 7D 7D 7D 7D
9193 7B 7B 7B 7B    303        DB      $7B,$7B,$7B,$7B,$7D,$7D,$7D,$7D
9197 7D 7D 7D 7D
919B 7B 7B 7B 7B    304        DB      $7B,$7B,$7B,$7B,$7D,$7D,$7D,$7D
919F 7D 7D 7D 7D
91A3 7B 7B 7B 7B    305        DB      $7B,$7B,$7B,$7B,$7D,$7D
91A7 7D 7D
91A9                306
91A9                307  BGDATA2:
91A9 7D 7D 7D 7D    308        DB      $7D,$7D,$7D,$7D,$7B,$7B,$7B,$7B
91AD 7B 7B 7B 7B
91B1 7D 7D 7D 7D    309        DB      $7D,$7D,$7D,$7D,$7B,$7B,$7B,$7B
91B5 7B 7B 7B 7B
91B9 7D 7D 7D 7D    310        DB      $7D,$7D,$7D,$7D,$7B,$7B,$7B,$7B
91BD 7B 7B 7B 7B
91C1 7D 7D 7D 7D    311        DB      $7D,$7D,$7D,$7D,$7B,$7B
91C5 7B 7B
91C7                312
91C7                313  ; CHARACTER WORK
91C7                314
91C7                315  CHRWORK:
91C7 0A 0A 01 01    316        DB      10,10,1,1
91CB 14 0F FF 01    317        DB      20,15,-1,1
91CF 05 05 01 FF    318        DB      5,5,1,-1
91D3 0F 0F 00 01    319        DB      15,15,0,1
91D7                320
91D7                321  ; VRAM WORK AREA
91D7                322  ;  30*29
91D7                323
B000                324        ORG     $B000
B000                325  VRAM3:
B000                326        DS      1024
B400                327  VRAM2:
B400                328        DS      1024
B800                329  VRAM1:
B800                330        DS      1024
BC00                331  VRAM0:
BC00                332        DS      1024
```

▶ 全機種共通システムインデックス ◀

＊以下のアプリケーションは，基本システムであるS-OS“MACE”またはS-OS“SWORD”がないと動作しませんのでご注意ください。

（で）のショートプロぱーてぃ――その42

PCMステレオ化大作戦！

Komura Satoshi　古村　聡

暖かい春までもう少し。今月号のショートプログラムは，ちょっと渋めに実用プログラム２本を紹介します。楽しく便利なパソコンライフを送るために活用するか，もっとよりよいものに改良するか。いずれにしても楽しく使ってくださいね。

illustration : T.Takahashi

今月の１本目に掲載するプログラムは，X68000のAD PCMを疑似的にステレオにするプログラムであります（ありゃ，タイトルそのまんまだ）。

X68000のAD PCM，サンプリング音源ってマイクから取った音をそのまま出せるから便利なんですけど，その音って基本的にモノラルだから，つながってるスピーカとかヘッドホンがステレオでも，左右両方から同じ音を出すか，左右どちらかしか出すことができなかったんですよね。

そいつを知恵と勇気とソフトウェアで乗り越えてしまうプログラムです。愛はハードウェアを超えるか？　さぁ，御用とお急ぎでない人は，よってらっしゃい見てらっしゃい（しまった今月はいつものヨタ話がない）。

ソフトで実現ステレオPCM

さて，さっそくプログラムの解説にいってしまいましょう。愛知県の秋山さんによる，X68000のAD PCM音源でステレオ再生に挑戦するPCMST.Sです。どうぞ！

PCMST.S&PCMSTDATA.C for X68000
（要XC ver.2.0以上，アセンブラ，リンカ，AD PCM-PCM変換ツール）

愛知県　秋山嗣晴

このプログラムは１チャンネルしかなく，左右のみか両方同じ音しか出せないX68000のAD PCM音源を使って疑似的にステレオ再生させるためのものです。すべてソフトウェアのみで行っているので，ハードなどの拡張はいっさい必要ありません。ただし，そのために音質はそれなりであることを了解しておいてください。

このプログラムを実行するにはこのリストを実行ファイルにするためのアセンブラ，リンカのほかにAD PCM-PCM相互にデータを変換可能なツール，たとえば，Z-MUSIC付属のツールZVT.Xなどが必要になります。

このプログラムはアセンブラのソースリストの形で書かれています。リスト１，リスト２をエディタで打ち込み，PCMST.S，PCMSTDATA.Cという名前でセーブしたあとで，リスト１を，

```
A>AS PCMST.S
A>LK PCMST.O
```

リスト２を，

```
A>CC  /Y  /W  PCMSTDATA.C
```

としてコンパイルすると実行ファイルPCMST.X，PCMSTDATA.Xができあがります。PCMST.Xがステレオ再生プログラム，PCMDATA.XはPCMデータ左右結合ツールです。アセンブラ，リンカは「X68000 Programing Series #1 X68k Develop」に収録されているHAS.X，HLK.Xを使ってもかまいません。その場合は，ASをHASに，LKをHLKに変えてください。

さて，このプログラムを実行するにはステレオ演奏するためのデータが必要になります。

PCMSTでステレオ再生を行うには，右用のAD PCMデータファイルと左用のAD PCMデータファイルを用意し，それを，右・左・右・左……と交互に並び替えてステレオデータに変換しなければなりません。

まず，ZVT.Xなどを使ってステレオ再生したい音の右チャネル，左チャネルのファイルを用意します。次にデータを加工しやすくするために，AD PCMデータからPCMデータに変換し，PCMSTDATA.Xで結合させます。さらに，結合させたPCMデータを，もう一度ADPCMファイルに変換し直すことで，データを作れます。

PCMSTDATA.Xは，

```
PCMSTDATA.X 右入力ファイル 左入力ファイル　MODE
```

と使います。MODEはサンプリング周波数のパラメータで，

```
mode= 0  3.9kHz
      1  5.2kHz
      2  7.8kHz
      3 10.4kHz
      4 15.6kHz
```

です。

ステレオデータの作り方は，リスト３にデータを作るためのバッチファイルを掲載します。参考にしてください。

そしていよいよ音データの再生です。

PCMST.Xは，

```
PCMST.X ファイル名　MODE
```

という形式で使います。具体的には，

```
A>PCMST.X SAMPLE1.PCM 4
```

という具合に使います。MODEはPCMSTDATA.XのMODEと同じです。MODE指定は必ず入れてください。

野心的な，AD PCM音源ステレオ再生プログラムですね。さすがにリアルタイム……というわけではありませんが，まさか，ショートプロでこういうプログラムをとりあげることになるとは思いませんでしたよ，わたしゃ。

先ほどもいったように，通常，X68000ではAD PCM音源は基本的にモノラルであるため，サンプリングされた音のデータを出力するのに左右のスピーカに同じ音を出させるか，あるいは右，左どちらかのスピーカからだけ出力を行います(左右の切り替えはできます)。そのサンプリングデータを再生している間，左右の出力は出力前に指定し，切り替えないで使うのが普通です。

こういう事実をふまえ，このX68000のAD PCM音源のデータを右用，左用と作り，サンプリングデータを再生するときに，左右を耳にわからないくらい高速に出力を切り替えるのです。右のスピーカをオンにするときには右のデータを，左のスピーカをオンにしているときには左のデータを出力すればあたかもステレオで音を再生しているかのようになるという理論によって作られたものです。リストを見ればわかりますが，AD PCMで再生中に左右の出力切り替えをするために，PCM8と同じようにDMAの継続動作モードと転送割り込み終了を使用し，データ転送終了時の割り込みで左右の出力を切り替え，次にDMAが転送するデータの設定を行っているのですね。かなり力技1本！　てな感じだけど，それなりに聴こえてくるからすごいなあ。

秋山さんはサンプルデータではいろいろなデータをいっしょにつけてくださったのですが，特にビールのコマーシャルで，ビールを注ぐ音「コポコポコポ……」という音が本当に奥行きを持って聴こえてくるのでびっくりしてしまいました。ただ，そのあと，私もいろいろデータを作って試してみたのですが，データがステレオに聴こえるかどうか，かなりの相性があるようですね。作者の秋山さんはプログラム自体はフリーウェアにする，ということなので，皆さんもぜひいろいろなデータを作って試してみてください。

ところでこのPCMST.Xを使うとデータ再生中にノイズが発生してしまいます。これは右・左の出力を切り替えるときに交互に音をオン/オフするためと，データを左右結合させるために図1のようにそれぞれの音のレベルが違うため，ブチブチとノイズが発生してしまうようなのです。この改良もどなたかやってみませんか？　音の立ち上がりと立ち下がりを強制的に音量0にするなどすれば，防げそうな気もするのですが……。お待ちしています。

環境は守るのではない，作るのだ

さぁて，今月はガンガンいっちゃいますよ。続いて2本目のプログラムは三重県の中山さんの作品でTITLE.Cです。どうぞ。

TITLE.C for X68000

(要XC ver.2.0以上)

三重県　中山剛志

このプログラムはCのプログラムをもっと快適に作るために，エディタ，コンパイラ，デバッガを有機的に結合させた統合環境ライクな環境を作るためのプログラムです。

MS-DOSマシンのTurboPASCALやTurboCってコンパイラを見たことありますか？　これってエディタとコンパイラとデバッガが一体化してて，すごく便利なコンパイラなんですよね。たとえばエディタ画面でプログラムのリストを打ち込んでるとしますよね。そこでRUNボタン(PC-9801だとCTRL＋F9のキー)を押すとワンタッチでプログラムをコンパイルして，実行して，終わったらもとのエディタの画面に戻ってくれるんです。そんな便利な環境を疑似的に実現するためのプログラムがこのTITLE.Cなのです。

図1　ノイズの原因

動かないよと思う前に(5)

★アドレスそれとも変数ですか？

変数の頭にあるはずの“&”がない。

C言語で“&”は変数の前に付いているとその変数の「アドレス」という意味になります。“&”がなければその変数の中身ですね。さて，この“&”があるべきときになかったらどうなるでしょう。たとえばアドレス65535を渡さなければならないところで，変数の中身100を渡してしまったとしたら……。そこに変数があると思って65335番地の中身を変えてしまったりするでしょうから，書き換えた場所がシステムなどで使っている場合，バスエラーになったり，最悪暴走してしまったりするのです。しかも文法としては成り立ってしまいますから，エラーは出ない……(型違いでWARNINGにはなる)。

★“&”はひとり，それとも2人？

さらに“&”は論理演算子AND(2月号のハンズで説明しましたよね)として使われます。ところが同じ条件式のAND，たとえば，a==1であって，かつ(AND)b==2のとき……というときのANDも同じように“&”を使って書くのです。

```
if ((a==1)&&(b==2))
```

のつもりで，

```
if ((a==1)&(b==2))
```

と書いてしまうのはまだしも，a & bをa && bと書いてしまったら文法的にも正しいので，WARNINGさえ出てきません。でも結果はおかしい。ということで“&”を使うときには“&”を忘れないように，ひとつか2つかを絶対間違えないようにね。うう，くわばら。

このプログラムはCのソースリストの形で書かれていますので，実行形式にするにはCコンパイラが必要です。ソースプログラムをエディタなどで入力して，XC ver. 2.0なら，

A>CC /W TITLE.C

としてコンパイルしてください。

それから，まず，環境変数で指定されているディレクトリ(AUTOEXEC.BATのファイルにPATH=……と書かれたディレクトリがいくつかありますよね。そのディレクトリです)か，できあがったプログラム，TITLE.Xのあるディレクトリと同じディレクトリにCOMMAND.X, SCD.X,SCD.CNF,SCD.HLP,PRINT.X,ED.Xを入れておいてください。これで準備はOKです。起動は簡単で，

A>TITLE

で実行できます。

で，このTITLE.Xの使い方はプログラムを実行すると画面にディレクトリの内容とその中の拡張子が“～.C”となっているものを表示してくれます。編集したいプログラムの名前を入力してください。画面に表示されていないプログラム名でもかまいません。ただし，拡張子の“.C”は自動的に付きますので入力しないでください。だからTEST.Cというプログラムを作りたいときにはTESTと入力すればいいことになります。

プログラム名を入力すると画面写真みたいなメニュー画面に変わります。あとはこのメニューでしたいことを書いてある数字を選んで入れるだけ。

1を押すとエディタ，ED.Xが立ち上がります。もし一度でもコンパイルしてからエディタを立ち上げると，そのときのコンパイラのエラーメッセージなどを表示します。プログラムの編集を行うには[ESC][V]キーを押してください。もう一度エラーを見たいときには[ESC][A]，または[ESC][D]と入力します。以後[ESC][A]または[ESC][D]を押すたびにエラーメッセージと編集プログラムが切り替わって表示されます。

2はコンパイルと実行です。コンパイルされていないプログラムを実行しようとするときは，これを使ってください。まず画面が変わってコンパイルを実行するかどうか聞いてきます。実行したければ小文字で“y”，コンパイラに渡すコマンドスイッチを変更したければ“s”，メニューに戻りたければそのほかのキーを押してください。“s”を押すとスイッチの入力を要求してきますのでスイッチのみ入力してください。CCやファイル名は入力しないでください。一度変更したスイッチは，もう一度変更されるまで変わりません。“s”はスイッチを替えることはできますが，出力されるプログラム名の変更などはできません。

3は，コンパイルされてすでにできている実行ファイルを実行します。

4はデバッガを実行します。このときCのソースコードでデバッグするには，実行ファイルだったらコンパイル時に“/Ns”オプションを付けたときだけです。“/Ns”をコンパイラのスイッチに付けなかった場合にはバイナリレベル(というかアセンブラレベルでの)デバッグになってしまいます。

5はプログラムのソースリストをプリンタに印字します。

6は編集するプログラムを変更したいときに使います。新しく編集するプログラムのファイルネームを入力してください。

7でこのプログラムを終了します。

なお，エディタやデバッガの使い方は，マニュアルを参照してくださいね。

てぇわけで，統合環境ライクな環境ができあがりました。まだ，ちょっとエディタから直接実行とかはできないけど(ちょっと無理かな？　ED.Xを書き換えなきゃならないかもしれない)，なかなか便利な環境であります。すでにmicroEMACSとGCCなんかでは，こういうことを実現しているマクロもあるらしいのですが，私のようにmicroEMACSが苦手の人としては嬉しいですよねぇ，やっぱり。

私は，フリーウェアのEDT.Xってエディタが好きなんで，それ向けに改造してしまおうと思ってます。それから，コンパイル&実行と再実行をひとつにまとめて……(ソースと実行ファイルのタイムスタンプを比べればできますよね。MAKE.Xを使ってもいいけど)。比較的改造しやすいプログラムだと思いますので，皆さんも使うコンパイラを変えるとか，プリントアウトのオプションを変えてみるとかいろいろと改造して，いちばん使いやすい環境を作ってみてくださいね。

リスト1　PCMST.S

```
 1: ***************************************************
 2: *
 3: *       ADPCMステレオ再生プログラム
 4: *
 5: *               PCMST.X
 6: *
 7: ***************************************************
 8:
 9:         .include        doscall.mac
10:         .include        iocscall.mac
11:
12:         .text
13:         .even
14:
15: start:
16:         lea.l   mysp,sp
17:
18:         suba.l  a1,a1
19:         iocs    _B_SUPER
20:         move.l  sp,super
21: arg:                                    *ファイルネーム読みこみ
22:         lea.l   fna,a0
23:         add.l   #1,a2
24:         tst.b   (a2)
25:         beq     arg_err
26:         cmp.b   #' ',(a2)
27:         beq     spetoba
28:         cmp.b   #$09,(a2)
```

```
 29:         beq     spetoba
 30: arg_loop:
 31:         cmp.b   #' ',(a2)
 32:         beq     arg_end
 33:         cmp.b   #$09,(a2)
 34:         beq     arg_end
 35:         move.b  (a2)+,(a0)+
 36:         bra     arg_loop
 37: spetoba:
 38:         add.l   #1,a2
 39:         bra     arg_loop
 40: arg_end:
 41:         clr.b   (a0)
 42:
 43: arg2:                                   *mode読みこみ
 44:         add.l   #1,a2
 45:         tst.b   (a2)
 46:         beq     arg_err
 47:         cmp.b   #' ',(a2)
 48:         beq     spetoba2
 49:         cmp.b   #$09,(a2)
 50:         beq     spetoba2
 51: arg_a:
 52:         move.b  (a2)+,d0
 53:         cmp.b   #' ',(a2)
 54:         beq     arg_end2
 55:         cmp.b   #$09,(a2)
 56:         beq     arg_end2
 57:         tst.b   (a2)
 58:         bne     arg_err
 59:         bra     arg_end2
 60: spetoba2:
 61:         add.l   #1,a2
 62:         bra     arg_a
 63: arg_end2:
 64:         clr.b   (a0)
 65: mode:                                   *modeから周波数、クロックを決める
 66:         sub.b   #'0',d0
 67: m_0:
 68:         cmp.b   #0,d0
 69:         bne     m_1
 70:         move.b  #0,samp
 71:         move.b  #$80,clk
 72:         move.l  #10,chg
 73:         bra     foread
 74: m_1:
 75:         cmp.b   #1,d0
 76:         bne     m_2
 77:         move.b  #4,samp
 78:         move.b  #$80,clk
 79:         move.l  #10,chg
 80:         bra     foread
 81: m_2:
 82:         cmp.b   #2,d0
 83:         bne     m_3
 84:         move.b  #0,samp
 85:         move.b  #$0,clk
 86:         move.l  #20,chg
 87:         bra     foread
 88: m_3:
 89:         cmp.b   #3,d0
 90:         bne     m_4
 91:         move.b  #4,samp
 92:         move.b  #$0,clk
 93:         move.l  #20,chg
 94:         bra     foread
 95: m_4:
 96:         cmp.b   #4,d0
 97:         bne     arg_err
 98:         move.b  #8,samp
 99:         move.b  #$0,clk
100:         move.l  #40,chg
101: foread:                                          *ファイル読みこみ
102:         clr.w   -(sp)
103:         pea.l   fna
104:         DOS     _OPEN
105:         add.l   #6,sp
106:         bsr     err
107:         move.l  #156000,-(sp)            *読み込むファイルサイズ
108:                                          *ここと fbufを変更すると
109:         pea.l   fbuf                     *もっと大きなファイルが
110:                                          *読み込めます
111:         move.w  d1,-(sp)
112:         DOS     _READ
113:         lea.l   10(sp),sp
114:         move.l  d0,count
115: fclose:
116:         move.w  d1,-(sp)
117:         DOS     _CLOSE
118:         add.l   #2,sp
119:
120: wari_set:                                        *割り込み設定
121:         move.l  #fbuf,add
122:         move.l  #$e840e5,a1
123:         move.b  #$6c,(a1)
124:         move.l  #$0001b0,a1
125:         move.l  #wari_pro,(a1)
126:
127: adpcm_stop:                                      *ADPCMの停止
128:         move.l #$e92001,a5
129:         move.b #1,(a5)
130:
131: adpcm_sample:                           *サンプリングレートの設定
132:         move.l  #$e9a005,a0
133:         move.b  samp,(a0)
134:
135: adpcm_outsel:                                    *出力左オン
136:         move.l  #$e9a007,a0
137:         clr.b (a0)
138:         move.b #3,(a0)
139:
140: adpcm_clksel:                                    *クロック設定
141:         move.l #$e90001,a0
142:         move.b #$1b,(a0)
143:         move.l #$e90003,a0
144:         move.b clk,(a0)
145:
146:         move.l #$e840c0,a4
147: clear_flag:                                      *DMAのCSRクリア
148:         move.b #$ff,(a4)
149: dma_setup:                                       *DMAの設定
150:         move.b #$80,4(a4)
151:         move.b #$32,5(a4)
152:         move.b #4,6(a4)
153:         move.b #8,7(a4)
154:         move.b #8,$2d(a4)
155:         move.b #5,41(a4)
156:         move.b #5,49(a4)
157:         move.l add,12(a4)
158:         move.l  chg,d0
159:         move.w  d0,10(a4)
160:         add.l   d0,add
161:         move.l #$e92003,20(a4)
162:
163: adpcm_start:                                     *ADPCMスタート
164:         move.b #2,(a5)
165:
166: dma_start:
167:         or.b     #$80,7(a4)                      *DMAスタート
168:
169:         move.l  add,$1c(a4)                      *次の転送の設定
170:         move.l  chg,d0
171:         move.w  d0,$1a(a4)
172:         add.l   d0,add
173:         move.b  #5,&39(a4)
174:
175:         or.b     #$40,7(a4)              *DMAコンティニューモード
176:
177: wait_complete:                                   *DMA,AD PCMが
178: w1:                                              *動作終了まで待つ
179:         move.b (a4),d0
180:         and.b #$90,d0
181:         bne w2
182:         move.b (a5),d0
183:         and.b #$80,d0
184:         beq w1
185: w2:
186:
187: adpcm_stop2:                                     *ADPCMの停止
188:         move.b #1,(a5)
189:
190: clear_flag2:                                     *DMAのCSRクリア
191:         move.b #$ff,(a4)
192:
193:         move.l  #$e840e5,a1                      *割り込みベクタをもどす
194:         move.b  #$6a,(a1)
195:
196:         move.l  super,a1
197:         iocs    _B_SUPER
198:
199:         DOS     _EXIT
200:
201: arg_err:
202:         pea     err_mes_arg
203:         DOS     _PRINT
204:         pea     usage
205:         DOS     _PRINT
206:         add.l   #8,sp
207:         DOS     _EXIT2
208: err:
209:         tst.l   d0
210:         bmi     error
211:         move.l  d0,d1
212:         rts
213: error:
214:         pea     err_mes_fopen
215:         DOS     _PRINT
216:         pea     usage
217:         DOS     _PRINT
218:         add.l   #8,sp
219:         DOS     _EXIT2
220:
221: wari_pro:                                        *割り込みのプログラム
222:         move.l  #$e840c0,a2
223:         move.l  chg,d1
224:         sub.l   d1,count                         *count<0なら終わる
225:         bcs     wari_end
226:
227:         move.b #$ff,(a2)                         *DMAの再設定
228:         move.b #8,7(a2)
229:         move.l  add,$1c(a2)
230:         add.l   d1,add
231:         move.w  d1,$1a(a2)
232:         move.b  #5,&39(a2)
233:
234:         or.b #$40,7(a2)
235:
236:         move.l #$e9a005,a2                       *L、Rきりかえ
237:         bchg.b  #1,(a2)
238:         bchg.b  #0,(a2)
239:
240:         rte
241: wari_end:
242:         move.b #$ff,(a2)                 *コンティニューモードをやめる
243:         move.b #0,7(a2)
244:         rte
```

```
245:
246:           .data
247: err_mes_arg:
248:           .dc.b    'パラメータが異常です',$0d,$0a,0
249: err_mes_fopen:
250:           .dc.b    'ファイルがありません',$0d,$0a,0
251: usage:    .dc.b    'STEREOPCM.X  filename mode[0=3.9KHz,
252:           .dc.b    '1=5.2KHz,2=7.8KHz,3=10.4KHz,
253:           .dc.b    '4=15.6KHz]',$0d,$0a,0
254:           .even
255:
256:           .bss
257: fbuf:     .ds.b    156000
258: fna:      .ds.b    40
259: super:    .ds.l    1
260: count:    .ds.l    1
261: add:      .ds.l    1
262: len:      .ds.w    1
263: samp:     .ds.b    1
264: clk:      .ds.b    1
265: chg:      .ds.l    1
266:           .even
267:
268:           .stack
269:           .even
270:
271: mystack:
272:           .ds.l    256
273: mysp:
274:           .end
275:
```

リスト2 PCMSTDATA.C

```
 1: /****************************************
 2:
 3:   PCMST.X用PCMデータ作成ツール
 4:
 5:           PCMSTDATA.X
 6:
 7:
 8: ****************************************/
 9: #include        <basic0.h>
10: #include        <basic.h>
11: void    paraerr();
12: void    fileerr();
13: void    err();
14: short   a[200],b[200];
15: int     fp1,fp2,fp3;
16: int     l,u,d;
17: /******** program start ********/
18: void
19: main(argc,argv)
20: int     argc;
21: char    *argv[];
22: {
23: if (argc!=5) paraerr();
24: d=*(argv[4]+1);
25: if (d!=0) paraerr();
26: d=*argv[4]-'0';
27: while (-1) {
28:         if (d==0) {u=40;break;}
29:         if (d==1) {u=40;break;}
30:         if (d==2) {u=80;break;}
31:         if (d==3) {u=80;break;}
32:         if (d==4) {u=160;break;}
33:         paraerr();
34: }
35: fp1=b_fopen(argv[1],"r");
36: if (fp1==-1) fileerr();
37: fp2=b_fopen(argv[2],"r");
38: if (fp2==-1) fileerr();
39: fp3=b_fopen(argv[3],"c");
40: if (fp3==-1) fileerr();
41: while   (-1){
42:         l=b_fread(a,2,u,fp1);
43:         if (l<u) break;
44:         b_fread(b,2,u,fp2);
45:         b_fwrite(&a[0],2,u/2,fp3);
46:         b_fwrite(&b[u/2],2,u/2,fp3);
47: }
48: b_fcloseall();
49: exit(0);
50: }
51: /****************************/
52: void    paraerr()
53: {
54: printf("パラメータが異常です¥n");
55: err();
56: }
57: /****************************/
58: void    fileerr()
59: {
60: printf("ファイルがありません¥n");
61: b_fcloseall();
62: err();
63: }
64: /****************************/
65: void    err()
66: {
67: printf("PCMMIX.X  入力ファイル1  入力ファイル2  出力ファイル  mode¥n")
;
68: printf("        (mode [0=3.9KHz,1=5.2KHz,2=7.8KHz,3=10.4KH
z,4=15.6KHz])¥n");
69: exit(1);
70: }
```

リスト3 PCM_MAKE.BAT

```
zvt -c %2 t2.tmp
pcmstdata t1.tmp t2.tmp t3.tmp 4
zvt -a t3.tmp %3
del t1.tmp
del t2.tmp
del t3.tmp
```

リスト4 TITLE.C

```
 1: #include        <stdio.h>
 2: #include        <stdlib.h>
 3: #include        <conio.h>
 4: #include        <basic0.h>
 5: #include        <basic.h>
 6: #include        <string.h>
 7: static  unsigned char   fina[40],k[40],p[50],sc[50],pr[5
0];
 8: int     no,cou;
 9: int     makefina();
10: int     men();
11: void     cmen();
12:
13: static  unsigned char   menu[10][40]={
14: "ソースプログラムの編集  " , "コンパイルして実行          " ,
15: "実行                    ","デバッグ                    " ,
16: "ソースプログラムの印字  " , "別のプログラムを編集する",
17: "終了                    " };
18:
19: static  unsigned char   strtmp0[258],strtmp1[258],strtmp
2[258];
20: /******** program start ********/
21: void
22: main()
23: {
24:         b_init();
25:         screen(1,2,1,1);
26:         console(0,29,0);
27:         width(96);
28:         b_csw(1);
29:
30:         cls();
31:         makefina();
32:         cls();
33:         while (men() != '7' ){
34:                 if ( no == '6' ){
35:                         cou=0;
36:                         cls();
37:                         makefina();
38:                         cls();
39:                 }
40:                 color(2);
41:                 printf("¥nHit Any Key!¥n");
42:                 getch();
43:                 width(96);
44:                 console(0,29,0);
45:         }
46:         b_exit(0);
47: }
48: /****************************/
49: int     makefina()
50: {
51:         int ai;
52:
53:         color(3);
54:         system("dir /w");
55:         color(1);
56:         system("dir /w *.c");
57:         color(2);
58:         b_input("編集するプログラム名を入力して下さい>",sizeof(fina)
,fina,-1);
59:
60:         strcpy(p,"ed ");
61:         strcat(p,fina);
62:         strcat(p,".c");
63:
64:         strcpy(sc,"scd ");
65:         strcat(sc,fina);
66:         strcat(sc,".x");
67:
68:         strcpy(pr,"print /B4 /N /T ");
69:         strcat(pr,fina);
70:         strcat(pr,".c");
71: }
72: /****************************/
73: int     men()
74: {
75:         int i;
```

```
 76:
 77:         cls();
 78:         color(3);
 79:         printf("編集プログラム名:");
 80:         color(10);
 81:         printf("%s.c¥n",fina);
 82:         color(1);
 83:         for (i=0;i<=6;i++){
 84:                 locate(40,i*2+8);
 85:                 printf("%d",i+1);
 86:                 printf(":");
 87:                 printf(menu[i]);
 88:                 printf("¥n");
 89:         }
 90:         printf("Input No.¥n? ");
 91:         no=getch();
 92:         cls();
 93:         switch (no){
 94:                 case '1':
 95:                         if(cou == 0){
 96:                                 system(p);
 97:                                 break;
 98:                         }
 99:                         else{
100:                                 system("ed err");
101:                                 break;
102:                         }
103:                 case '2':
104:                         cmen();
105:                         break;
106:                 case '3':
107:                         color(3);
108:                         system(fina);
109:                         break;
110:                 case '4':
111:                         color(3);
112:                         system(sc);
113:                         break;
114:                 case '5':
115:                         system(pr);
116:                         break;
117:                 case '6':
118:                         break;
119:                 default:
120:                         break;
121:         }
122:
123:         return(no);
124: }
125: /*****/
126: void cmen(void)
127: {
128:         int i,no;
129:         char ans;
130:         static char sw1[100];
131:         static char sw2[100];
132:
133:         color(2);
134:         printf("cc %s %s.c",sw1,fina);
135:         color(3);
136:         printf(" でコンパイルします¥n");
137:         printf("実行(y) スイッチの変更(s) 中止(その他)¥n");
138:         ans=getch();
139:         if((ans != 'y') && (ans != 's'))
140:                 return;
141:
142:         if(ans == 's'){
143:                 strcpy(sw1,"");
144:                 b_input("スイッチを入力して下さい:",sizeof(sw1),
sw1,-1);
145:         }
146:
147:         strcpy(sw2,"cc ");
148:         strcat(sw2,sw1);
149:         strcat(sw2," ");
150:         strcat(sw2,fina);
151:         strcat(sw2,".c");
152:         strcat(sw2," > err");
153:         cou++;
154:         printf("%s¥n",sw2);
155:         system(sw2);
156:         system("type err");
157:         cls();
158:         color(3);
159:         system(fina);
160: }
```

ぱーてぃハンズ(5)

しっかり動かす

2月号のリストでは，左右の移動のほかにジャンプができるようになってるんですよね。ジャンプはジョイスティックを上に押し倒す，つまり8の方向にすることでできます。斜めジャンプは7と9の位置に倒すことでできます。

さて，ここでその前の解説を思い出して(あるいは見直して)ください。ジョイスティックを倒したことは，なにをどうするとわかるんでしたっけ？

そう，a＝stick()とすると，aの中身がスティックの方向に応じて変わってるんですよね。たとえば，上なら8，横なら6という具合に。で，先月のさらに前のリストではその中身を見るのに，

　if (stick(1)＝4)～
　if (stick(1)＝6)～

として調べていたんでしたよね。

それじゃぁ，ということでこれをジャンプの場合や斜めジャンプのときにも同じようにif文でずらずらっと，

　if (stick(1)＝4)～
　if (stick(1)＝6)～
　if (stick(1)＝7)～ ←左ジャンプ
　if (stick(1)＝8)～ ←ジャンプ
　if (stick(1)＝9)～ ←右ジャンプ

こんなふうに追加してしまえば，当然，いままで左右に動いていたのと同じようにジャンプをさせることができるわけですよね。うん，正しい。

ところが，こうやって書いてしまうとあまりきれいではないですよね。で，ここで登場するのがswitch文です。

●swicth～case～[default] ～endswicth

　switch <式0>
　　case <式1>：<文1>
　　case <式2>：<文2>
　　　　：
　　case <式n>：<文n>
　　[default: <文n>]
　endswitch

なんだかややっこしい説明が載ってるんですけど，簡単にいってしまうとswitch文というのは，たくさんあるif文をまとめてしまったものなんです。swicth<式1>と書いてある式1に，たとえば，

　if (stick(1)＝なんとか)

って書いてある同じif文がたくさんあったら，そのifの()のなかの「なんとか」という式を書いてやると，あとはcaseのあとにその条件を書いておくことで，たくさんのif文の代わりをしてくれるんです。今回の場合ならさっきのリストを，

　switch (stick(1))
　　case 4：～
　　　break;
　　case 6：～
　　　break;
　　case 7：～
　　　break;
　　case 8：～
　　　break;
　　case 9：～
　　　break;
　endswitch

とすっきりさせることができるんです。かなり見やすいですよね，こちらのほうが。

そうそう，breakというのは次のcaseの直前に付ける決まり文句だと思ってください。本当はちょっとした意味があるんですが，まあ，たいした意味ではありません(いいのかな，ここまでいい切って……。知りたい人はマニュアルで調べてね)。

それから，歩きながらパンチしないようにということですが，これは簡単にできます。kという変数を作ります。kickのkだと思ってください。で，このkの中身が0ならパンチもキックもしていない，としましょう。

ということは，キックやパンチをしたときにkに0でない数字を入れればいいんですよね。パンチやキックをしようとしたときというのは，ジョイスティックのボタンが押されているはずですから，stick(1)が1や2になっているわけです。

　if (stick(1)＝2)
　if (stick(1)＝4)

のときにkに2や4にしています。え，なんで1でなくて2とか4が入っているのかって？これはパンチやキックをしてから，2または4回ループが回ってくるまで体が動かなくなる「スキ」を作ろうと思ったんですよね。これは実際使うかどうかわからないんですけどね。そんなところです。

で，なんだか今月は脱線してしまいましたけど，来月こそは本当にいよいよ，2人キャラを出しましょう。

来月はまたリストが出ますよ。さあ，がんばらなきゃ！

in Z80's Bar

illustration : T.Takahashi

今月はマス目のサイズの変更と，爆弾の個数を自由に設定できるようにします。ちょっとした拡張ですので，力のある人はすでにやってしまったかもしれませんね。自分のレベルに合わせて，さらに楽しく爆弾掃除人を遊んでください。

必殺！爆弾掃除人(発展編)

Kaneko Shunichi **金子 俊一**

カランコロ～ン♪

長老(以下老)：お一来おった，来おった。

ようこ(以下Yo)：待ってたわ～。

源光(以下光)：ふっ，僕って人を待たせるからな。

マスター(以下M)：にくいね，このぉ。

老：あいかわらずおめでたい奴らじゃのう。

光：いわれるまでもありませんね。どうせプログラムを作れっていうんでしょ。

Yo：よくわかってるわね。

光：金さんの桜吹雪よかワンパターンですよ。

老：VSOPってか。

M：べりー・すぺしゃる・ワン・パターンね。

光：いまどき小学生でもそんなことはいいませんよ。

Yo：そうよね。

光：それはともかく，ようこさんっ！

Yo：はい？

光：そのワンレン・ボディコン・ハイヒールにピンクの扇子はなんですか？

Yo：決まってるじゃない，ジュリアナトーキョーに行くのよ。

リスト1

```
C6B6 CD 24 C0 11 39 C7 CD E5 : 74
C6BE 1F 11 51 C7 CD E5 1F 3A : 53
C6C6 1A C4 CD C1 1F 21 0B 01 : B8
C6CE CD 29 C7 38 F8 FE 02 38 : 25
C6D6 F4 FE 11 30 F0 32 1A C4 : 33
C6DE 47 11 5D C7 CD E5 1F 3A : 87
C6E6 1B C4 CD C1 1F 21 0B 02 : BA
C6EE CD 29 C7 38 F8 FE 02 38 : 25
C6F6 F4 FE 11 30 F0 32 1B C4 : 34
C6FE 4F AF 81 10 FD B7 20 01 : 64
C706 3D 47 11 69 C7 CD E5 1F : 96
C70E 3A 1C C4 CD C1 1F 21 0B : F3
C716 03 CD 29 C7 38 F8 B7 28 : CF
C71E F5 B8 30 F2 32 1C C4 E1 : C2
C726 C3 03 C0 CD 1E 20 11 80 : 22
C72E C7 CD D3 1F 11 8B C7 CD : B6
---------------------------------
SUM: 32 83 FA DC FF 95 D3 D5 7F18

C736 B5 1F C9 4D 49 4E 45 53 : 19
C73E 57 45 45 50 45 52 20 4F : 37
C746 50 54 49 4F 4E 20 53 45 : 42
C74E 54 0D 00 20 4C 45 4E 47 : A7
C756 54 48 20 3D 20 24 00 20 : 5D
C75E 57 49 44 54 48 20 20 3D : FD
C766 20 24 00 20 4D 49 4E 45 : 8D
C76E 53 20 20 3D 20 24 00 20 : 34
C776 4C 4F 43 41 54 45 20 3D : 15
C77E 20                      : 20
---------------------------------
SUM: 3A E9 1E 3B 51 FB 94 2D 4B43
```

$C77F_H$～$C7CF_H$まで00_Hで埋める

光：へっ？

Yo：ほら。Oh!Xの編集室って泉岳寺にあるじゃない，遊びにいったついでにジュリアナトーキョーよ。

光：ってことはOh!X編集室にその格好でいくの？

Yo：そうよ。なんかへん？　ちゃんと扇子も持ってるし。

老：そんな格好でいったら，古村氏の雑巾がけのような伝説がOh!X編集室に残るじゃろうな。

光：と，ともかくプログラムでもなんでも作りますから，今日はここにいることにしましょう。ねっ，ようこちゃん。

Yo：それじゃあマインスイーパーの拡張をしてね。

やらせていただきます

光：何がやりたいの？

Yo：最低限，サイズを変えることは必要ね。それぐらいはできるでしょ。

光：よしきた，がってんだい。

老：今日はいやに素直に承知したもんじゃのう。

光：Z80's Barのカウンターをお立ち台にするくらいならかわいいですけどね，編集室で笑い者にされるのはさすがに忍びなくて。

M：同感。

Yo：何かいった？

M：いえ，なんにも。

光：さっそくコーディングしちゃいますね。カチャカチャ……。

老：そんないきあたりばったりで作れるんかいのう。

光：いきあたりばったりで拡張できるように，先月は思いっきり苦労したんですよ。カチャカチャ……。

Yo：プログラム長かったもんねぇ。

光：先月号のプログラムは，動くプログラムを一度コーディングしてから，変更しやすいように作り直したものなんですよ。カチャカチャ……。

老：どうりで力技なアルゴリズムだったわけだ。

Yo：まあ，いいじゃない。「動くプログラムはすべて美しい」って言葉もあるくらいだから。

光：でーきたっと。

出来栄えは？

Yo：早かったわね。

光：先月の苦労が報われたって感じかな。使い方は，先月号のダンプリストに続いて，今月号のリスト1を打ち込んでもらえるだけでOKです。

M：このプログラムで，どこが変わったのか教えてくださいよ。

光：縦，横ともに2～16の間で自由にサイズを変えられるようになりました。

老：最小が2×2，最大は16×16ってことじゃな。

M：爆弾の数はどうなりました？

光：1～(面積－1)個まで設定できます。だけど，2月号でもいったとおり，あんまり多くしすぎると，なかなか画面が出てこなくなりますよ。

M：何か目安になるようなものはあるんですか？

光：爆弾の数は最大でも面積の半分程度にしないと，クリアは難しいんじゃないかな。Windows版では，初級レベルで64マス中10個の爆弾だったのを参考にするといいでしょう。

老：もちろん，設定範囲をオーバーするようなことはできないようにチェックしてあるんじゃろ。

光：当然の助動詞ですね。

M：この設定画面にくるのはどうすればいいんですか？

光：先月のリザーブの中で，「O」というのがあったでしょ。

老：オプションの略じゃな。

光：別にCARBOYでもヤングバージョンでもよかったんですけどね。

Yo：そのボケを理解できる人は少ないんじゃない？

超絶技巧テクニック

M：今回のテクニックの目玉はズバリなんでしょう。

光：ありません。

M：そんなぁ。

光：目玉ってほどのもんじゃありませんけど，1行入力なんてのは面白いかもしれませんね。

老：S-OS“SWORD”のサブルーチンにあるやつじゃな。

光：ええ。

Yo：前に使ってみようとしたことがあったんだけど，うまくいかなかったのよ。

光：入力用のバッファを確保しました？

Yo：それなぁに？

光：バッファを80文字分確保しなくちゃいけませんよ。

Yo：だから暴走したのかな。

光：するかもしれない。

M：でも，アセンブラで文字列操作って面倒臭いんじゃありませんか？

光：何文字目に目的のデータがあるかがわかっていれば，なんとかなりますよ。

老：今回ではどんな具合に対処しておるのじゃ？

光：カーソルを表示することで，データを入力してほしい場所を明確にしました。

老：このカーソルは画面中を移動できるのじゃろ。端っこのほうにデータを書くやつがおったらどうするのじゃ。

光：現在の設定値が表示してある部分しか判断しないから大丈夫なんです。

Yo：そこにスペースとか数値以外のデータが書いてあった場合はどうするの？

光：もちろん，エラーのチェックはしてますから，エラーなら再度入力ということですね。

Yo：なるほど。

老：設定範囲を超えたエラーも再度入力させるのじゃな。

M：ほかにも使い道はあるんですか？

光：ディレクトリを表示して，ファイル選択なんてのも可能でしょうね。

Yo：エディタのWINNERなんかがやっている方法ね。

光：作り方がわかればとても便利でしょ。

Yo：なるほど。

ジェームス・ブラウン！

Yo：それじゃあ話がまとまったつぅことで，いってきまーす。

カランコロ～ン♪

光：あ～あ，いっちゃったよ。

M：あっ，そういえば今日は出張校正だからOh!X編集室には誰もいないんじゃないのかな。

光：ジュリアナも貸し切りパーティやってたりして。

M：そうなったら，ようこちゃん路頭に迷うだろうな。

老：いやあ，めでたし，めでたし。

光：なんか違う気がする。

—つづく—

リスト2

```
0000                1 ;   MINESWEEPER OPTION SET.
0000                2 ;
0000                3 ;    by Hikaru Minamoto
0000                4
0000                5
0000                6 OPTION  EQU     $C6B6
0000                7 LENGTH  EQU     $C41A
0000                8 WIDTH   EQU     $C41B
0000                9 MINES   EQU     $C41C
0000               10 LOC     EQU     $C41D
0000               11 CLS     EQU     $C024
0000               12 START   EQU     $C003
0000               13
0000               14 #2HEX   EQU     $1FB5
0000               15 #PRTHX  EQU     $1FC1
0000               16 #GETL   EQU     $1FD3
0000               17 #MSX    EQU     $1FE5
0000               18 #LOC    EQU     $201E
0000               19
0000               20
C6B6               21         ORG     OPTION
C6B6               22
C6B6 CD 24 C0      23         CALL    CLS
C6B9 11 39 C7      24         LD      DE,MES1
C6BC CD E5 1F      25         CALL    #MSX
C6BF               26         ;
C6BF 11 51 C7      27         LD      DE,MES2
C6C2 CD E5 1F      28         CALL    #MSX
C6C5 3A 1A C4      29         LD      A,(LENGTH)
C6C8 CD C1 1F      30         CALL    #PRTHX
C6CB               31 LE
C6CB 21 0B 01      32         LD      HL,$010B
C6CE CD 29 C7      33         CALL    GET
C6D1 38 F8         34         JR      C,LE
C6D3 FE 02         35         CP      2
C6D5 38 F4         36         JR      C,LE
C6D7 FE 11         37         CP      17
C6D9 30 F0         38         JR      NC,LE
C6DB 32 1A C4      39         LD      (LENGTH),A
C6DE 47            40         LD      B,A
C6DF               41         ;
C6DF 11 5D C7      42         LD      DE,MES3
C6E2 CD E5 1F      43         CALL    #MSX
C6E5 3A 1B C4      44         LD      A,(WIDTH)
C6E8 CD C1 1F      45         CALL    #PRTHX
C6EB               46 WI
C6EB 21 0B 02      47         LD      HL,$020B
C6EE CD 29 C7      48         CALL    GET
C6F1 38 F8         49         JR      C,WI
C6F3 FE 02         50         CP      2
C6F5 38 F4         51         JR      C,WI
C6F7 FE 11         52         CP      17
C6F9 30 F0         53         JR      NC,WI
C6FB 32 1B C4      54         LD      (WIDTH),A
C6FE 4F            55         LD      C,A
C6FF AF            56         XOR     A
C700               57 MAX
C700 81            58         ADD     A,C
C701 10 FD         59         DJNZ    MAX
C703 B7            60         OR      A
C704 20 01         61         JR      NZ,MAX2
C706 3D            62         DEC     A
C707               63 MAX2
C707 47            64         LD      B,A
C708               65         ;
C708 11 69 C7      66         LD      DE,MES4
C70B CD E5 1F      67         CALL    #MSX
C70E 3A 1C C4      68         LD      A,(MINES)
C711 CD C1 1F      69         CALL    #PRTHX
C714               70 MI
C714 21 0B 03      71         LD      HL,$030B
C717 CD 29 C7      72         CALL    GET
C71A 38 F8         73         JR      C,MI
C71C B7            74         OR      A
C71D 28 F5         75         JR      Z,MI
C71F B8            76         CP      B
C720 30 F2         77         JR      NC,MI
C722 32 1C C4      78         LD      (MINES),A
C725               79         ;
C725 E1            80         POP     HL ;DUMMY
C726 C3 03 C0      81         JP      START
C729               82
C729               83 GET
C729 CD 1E 20      84         CALL    #LOC
C72C 11 80 C7      85         LD      DE,BUF
C72F CD D3 1F      86         CALL    #GETL
C732 11 8B C7      87         LD      DE,BUF+$0B
C735 CD B5 1F      88         CALL    #2HEX
C738 C9            89         RET
C739               90 MES1
C739 4D 49 4E      91         DM "MINESWEEPER OPTION SET"
C73C 45 53 57
C73F 45 45 50
C742 45 52 20
C745 4F 50 54
C748 49 4F 4E
C74B 20 53 45
C74E 54
C74F 0D 00         92         DB      $0D,$00
C751               93 MES2
C751 20 4C 45      94         DM " LENGTH = $"
C754 4E 47 54
C757 48 20 3D
C75A 20 24
C75C 00            95         DS      1
C75D               96 MES3
C75D 20 57 49      97         DM " WIDTH  = $"
C760 44 54 48
C763 20 20 3D
C766 20 24
C768 00            98         DS      1
C769               99 MES4
C769 20 4D 49     100         DM " MINES  = $"
C76C 4E 45 53
C76F 20 20 3D
C772 20 24
C774 00           101         DS      1
C775              102 MES5
C775 20 4C 4F     103         DM " LOCATE = "
C778 43 41 54
C77B 45 20 3D
C77E 20
C77F 00           104         DS      1
C780              105 BUF
C780 00 00 00     106         DS      80
```

愛読者プレゼント

●プレゼントの応募方法●

とじ込みのアンケートはがきの該当項目をすべてご記入のうえ，希望するプレゼント番号をはがき右下のスペースにひとつ記入してお申し込みください。締め切りは1993年３月18日の到着分までとします。当選者の発表は1993年５月号で行います。また，雑誌公正競争規約の定めにより，当選された方はこの号の他の懸賞には当選できない場合がありますのでご了承ください。

1

ソフトプラン
☎08669(3)8686

キングス・ダンジョン

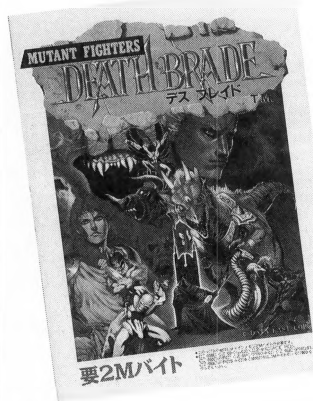
X68000用　5"2HD版
5,800円(税別)　**5名**

ダンジョンに侵入し，そこに棲む生物たちを殺戮して宝を奪っていく戦士たち。君はモンスターを召喚できる魔王に扮し，そいつらを蹴散らす。戦士たちはちゃんと思慮深く行動するし，パーティどうしの戦闘も見モノだぞ。

2

エス・ピー・エス
☎0245(45)5777

デスブレイド

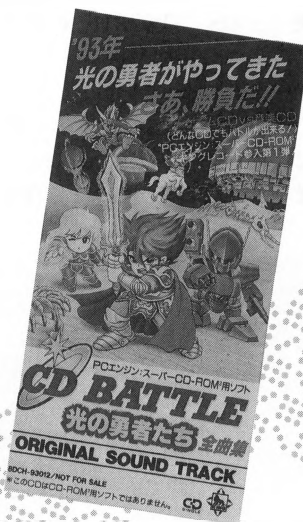
X68000用　5"2HD版
9,800円(税別)
3名

力だけがすべての世界。男に生まれたからにはそんな世界に浸りたい，でも体力ないから実体験は遠慮したい。そんな君でもこの「デスブレイド」をプレイするだけなら，叩かれる顔も叩く手も痛くもかゆくもない。

3

キングレコード
☎03(3945)2122

CD BATTLE 光の勇者たち

10名

CDどうしでバトルができる，PCエンジン用ソフト「CD BATTLE 光の勇者たち」。このゲームの音楽と効果音を収録したCDシングルをプレゼント。さらに，これをCDバトルに使用すると……。非売品です。

4

メディックス
☎03(3950)2222

MIRAGE System ロゴ入り袋＆ボールペン

15名

まずは「Model Stuff」で3D CG作成ソフト“MIRAGE System”シリーズを始動させている，メディックスさんから販促品をいただきました。袋とボールペン２本をセットにしてプレゼントします。

１月号モニタ当選者

M❶ピクノ＆モンタージュカード　(千葉県)酒元一幸　(愛知県)大塚竜志　(大阪府)尾上仁彦　M❷Multiword　(新潟県)保科康広　(埼玉県)鈴木真一　(静岡県)大山和紀　(敬称略)
以上の方々が当選しました。商品は順次発送いたしますが，入荷状況などにより遅れる場合もあります。

１月号プレゼント当選者

P❶ふしぎの海のナディア　(北海道)小林義孝　(東京都)石井清貴　堀義弘　P❷3D下敷き　(栃木県)鈴木広志　(茨城県)倉田泰幸　(埼玉県)小川純一　(東京都)井上綾子　(愛知県)石川淳二　渡邊哲　(兵庫県)浪越孝宏　(広島県)桐本順功　仁井内明　(敬称略)
以上の方々が当選しました。商品は順次発送いたしますが，入荷状況などにより遅れる場合もあります。

●編集部へのメッセージ

今月号の特集について

いちばん良かった記事

興味のなかった記事

これから載せてほしい記事内容

本誌以外にお読みのパソコン雑誌

期待している新作ソフト：
推薦理由：

最近買って気に入ったソフト：
推薦理由：

X68030の第一印象はどうでしたか？

あなたの愛機は(所有機種に○印をつけてください)　　　　ない
X1(マニアタイプ,C,D,F,G,twin) X1 turbo(model 10,20,30,40,II,III,Z,ZII,ZIII)
MZ-(80K/C, 1200, 700, 1500, 80B, 2000, 2200, 2500, 2861)
X68000(初代,ACE,PRO,PROII,EXPERT,EXPERTII,SUPER,XVI,Compact, HD)
X68030(CZ-500,CZ-510)
その他　　　　　　　　　　　　　　MIDI楽器(　　　　　　)
FD(　　基)　TAPE　QD　HD(　　MB)　MO　プリンタ(　　　　　　)

年齢　　歳　　　パソコン歴　　年　　　男・女　　　プレゼントNo.

Oh!X 愛読者アンケート

今年も５月号で読者特集「言わせてくれなくちゃだワ」を予定しています。つきましては愛読者の皆さんにアンケートのご協力をお願いすることになりました。どうか皆さんの本音をお聞かせください。よろしくお願いいたします。

<応募方法> 回答用紙に必要事項を記入のうえ，封書で下記の宛て先までお送りください。
<宛て先> 〒108 東京都港区高輪2-19-13 NS高輪ビル　ソフトバンク株式会社　Oh!X編集部 読者アンケート係
<締め切り> 1993年3月15日（当日消印有効）
<賞品> ご協力いただいた方のなかから抽選でポータブルCDプレイヤーを２名の方に，100名の方に記念品を差し上げます。

住所	〒 電話　（　　）	氏名	年齢　男・女
職業・勤務先 学校・学部・学年			

1．あなたの所有する機種（本体）にすべて○印をつけてください。ただし，２機種以上お持ちの方は，現在メインマシンとして使用している機種だけ◎をつけてください。
X1(マニアタイプ，C，D，F，G，twin)　X1turbo(model10，20，30，40，II，III，Z，ZII，ZIII)
X68000（初代，ACE，ACE-HD，PRO，PRO-HD，EXPERT，EXPERT-HD，PROII，PROII-HD，EXPERTII，EXPERTII-HD，SUPER，SUPER-HD，XVI，XVI-HD，Compact）　X68030/HD　MZ-（80K/C/1200，700，1500，80B，2000/2200，2500，2861）
PC-98（　　　），PC-88（　　　），エプソンPC-（　　　），FM（　　　），MSX（1，2，2＋，TurboR），AX（　　　）
IBM PC（　　　），Macintosh（　　　），AMIGA（　　　），ノートパソコン（　　　），ポケコン（　　　）
その他　ない

2．システム環境についてお聞きします。お持ちのハードウェアとソフトウェアの□に印をつけ，必要な項目にご記入ください
●X68000をお持ちの場合
□ディスプレイ　□メインメモリ（　　MB）　□数値演算プロセッサ　□SCSIボード　□光磁気ドライブ　□CD-ROM
□ハードディスク　内蔵（　　MB）　外付（　　MB）　□リムーバブルHD（　　MB×　枚）　機種名：
□プリンタ（カラー，モノクロ）　機種名：　　　メーカー名：
□ビデオプリンタ　□カラーイメージユニット　□カラーイメージスキャナ　□ハンディスキャナ（　　　）
□MIDIボード　□ビデオボード　□FAXボード　□拡張I/Oボックス　□サイバースティック　□ジョイスティック
□POLYPHON　□V70ボード　□MIC 68K
□モデム　機種名：　　　メーカー名：
□その他のハード
□SX-WINDOW ver.（　　）　□THE 福袋ver.2.0　□C compiler PRO-68K ver.（　　）　□OS-9/X68000
□BUSINESS PRO-68K（Kamikaze）　□CARD PRO-68K　□DATA PRO-68K　□CHART PRO68K　□Communication SX-68K
□SOUND（PRO-68K，SX-68K）　□Sampling PRO-68K　□MUSIC PRO-68K（[MIDI]）　□Musicstudio PRO-68K　□Mu-1
□CANVAS PRO-68K　□NEW PrintShop PRO-68K　□HyperWord　□MultiWord　□THUNDER WORD　□EasyPaint SX-68K
□Z'sSTAFF PRO-68K　□Matier　□C-TRACE（　　）　□サイクロン（　　）　□MIRAGE System Model Stuff　□DoGA･CGAシステム
□FIXER ver.4.0　□NAGDRV　□Z-MUSIC　□PressConductor PRO-68K　□Y300-A
●S-OSをお使いの方はお持ちのシステムを教えてください
□S-OS SWORD（使用機種：　　　）　□CP/M（使用機種：　　　）
□マシン語入力ツール（MACINTO-C，その他　　　）　□MAGIC（使用機種：　　　）
□エディタ（E-MATE，WINER，TED-750，EDC-T，その他：　　　）
□アセンブラ（REDA，ZEDA，ZEDA-3，WZD，OHM-Z80，その他：　　　）
□コンパイラ（FuzzyBASIC，SLANG，TTC，TTC＋＋，STACK，Small-C）　□インタプリタ（FuzzyBASIC，TTI，STACK）
□その他（　　　）

3．あなたは主としてパソコンをどのような用途で使用されていますか？　該当する項目にチェックしてください（複数選択可）
□ゲーム　□実務処理　□ワープロとして　□プログラミング　□パソコン通信　□CG　□音楽　□ビデオの制作
□各種機器の制御　□コンピュータ入門用として　□インテリアとして　□その他

4．現在お持ちのパソコンを次の項目について（○△×）の３段階でチェックしてください
1基本スペック・性能（　）　2使いやすさ（　）　3デザイン（　）　4コストパフォーマンス（　）
5OS/システム環境（　）　6プログラミング環境（　）　7ゲーム環境（　）　8ビジネス環境（　）
9グラフィック環境（　）　10音楽環境（　）　11周辺機器の充実度（　）　12メーカーサポート（　）
13ユーザーの活性度（　）　14個人的な満足度（　）

キリトリ線

5．プログラミング言語についてお聞きします。プログラムの開発に使っているものには◎，いちおう理解できる言語は○，まだよくわからないが関心はある言語には△，関心のないものは×を記入してください

□BASIC（　）　□C言語（　）　□アセンブラ（　）　□C＋＋（　）　□その他（　　（　））

6．常用されているフリーソフトウェアなどがありましたら教えてください

□microEMACS ver.（　）　□GCC ver.（　）　□NEMACS　□G＋＋　□HAS　□HLK　□DIS　□TeX　□LHA　□LZX
□APIC　□MAKI　□MAG　□PI　□MFGED　□PST　□PCM8　□MXDRV　□MDDRV　□DI　□TF　□ADDRV　□CONDRV
□その他（

7．MIDI楽器をお持ちですか

□MT-32/CM-32　□CM-64　□SC-55/CM-300　□CM-500　□D-10/20　□M1/EX　□DX-7　□TG100
□その他（機種名：　　メーカー名：　　）

8．Oh!Xについて

●Oh!Xを購入されたのは初めてですか？
□初めて買った　□以前からときどき買っている　□最近よく買っている　□ほとんど毎月買っている
●Oh!Xの付録ディスクをお持ちですか
□創刊８周年記念PRO-68K　□謹賀新年PRO-68K　□黄金週間PRO-68K　□創刊10周年記念PRO-68K
●Oh!Xの筆者で気に入っている人がいれば教えてください（２名まで）
筆者名（　　）　理由

9．次の１～10のうちからお好きなテーマを選んで（いくつでも可）番号を記入のうえ，自由に言いたいことをお書きください。ユニークなご意見，エピソードなどを期待しています

1パソコン界の動向または未来について　2ハードメーカー（シャープ）に関して　3こんなソフトを出してほしい
4 Oh!Xに関して　5あなたとパソコンの関係　6あなたのまわりのヘンなユーザー　7 X68000，100万台への野望
8あなたが期待する次世代のパソコン　9面白い話があるので書きたい　10とにかく言っておかねばならないことがある

キリトリ線

振替用紙

←点線から、きれいに切り取ってご使用ねがいます。

払込票

通常払込料金加入者負担

口座番号	東京 1 — 29307
加入者名	ソフトバンク株式会社
金額	億 千 百 十 万 千 百 十 円
払込人住所氏名	※
備考	受付局日付印

切り取らないで郵便局にお出しください。

記載事項を訂正した場合は、その箇所に訂正印を押してください。

払込通知票

通常払込料金加入者負担

口座番号	東京 1 — 29307	金額	億 千 百 十 万 千 百 十 円
加入者名	ソフトバンク株式会社	料金	払込み ／ 特殊 円
払込人住所氏名	※ （郵便番号　　　　）	備考	
		受付局日付印	

各票の※印欄は、払込人において記載してください。

この払込通知票は、機械で使用しますので、下部の欄を汚さないよう特に御注意ください。また、本票を折り曲げたりしないでください。　（郵　政　省）

【定期購読のご案内】

●定期購読のお申し込みは、この郵便振替用紙のみとさせて頂き、銀行振込・現金書留によるご入金は、ご遠慮下さい。

●受付締切は、

1日発売	：	発売日前月10日振込
8日発売	：	〃　15日振込
15・18日発売	：	〃　25日振込

です。

〈例〉 4月1日発売（Oh！PC 4月15日号）の場合、お振込の締切は3月10日です。

締切に間に合わなかった場合は、自動的に次号からの発送となります。

なお、すでに発売されているもの、また、お振込が締切に間に合わなかった月号のものは、定期購読ではお求めになれません。書店でご購入ください。

●定期購読誌のお届けは書店発売日より遅くなりますのでご了承下さい。

「発売日一覧」

◇毎月1・15日発売
Oh！PC

◇毎月18日発売
Oh！X
Oh！FM TOWNS
C MAGAZINE
Oh！Dyna

◇毎月8日発売
月刊情報処理試験
LANTIMES
THE WINDOWS
DOS／Vmagazine
UNIX USER

●月刊情報処理試験は93年1月号より定期購読料金を改訂させていただきます。お申し込みの際はご注意下さい。

切り取り線

切り取らないで郵便局にお出しください。

この欄は、加入者あての通信にお使いください。

送り先

お名前　フリガナ　　お電話

ご住所　フリガナ　〒

定期購読申込書

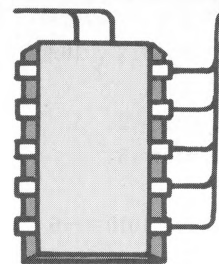
□Oh！PC	年間	（23回）	12,880円	（新規／継続 NO.　　）
□Oh！PC	6ヶ月	（12回）	6,720円	（新規／継続 NO.　　）
□Oh！X	年間	（12回）	7,200円	（新規／継続 NO.　　）
□Oh！FM TOWNS	年間	（12回）	7,440円	（新規／継続 NO.　　）
□C MAGAZINE	年間	（12回）	11,760円	（新規／継続 NO.　　）
□Oh！Dyna	年間	（12回）	9,120円	（新規／継続 NO.　　）
□月刊情報処理試験	年間	（12回）	9,360円	（新規／継続 NO.　　）
□月刊情報処理試験	6ヶ月	（6回）	4,680円	（新規／継続 NO.　　）
□LANTIMES	年間	（12回）	17,760円	（新規／継続 NO.　　）
□THE WINDOWS	年間	（12回）	11,760円	（新規／継続 NO.　　）
□DOS／Vmagazine	年間	（12回）	9,360円	（新規／継続 NO.　　）
□UNIX USER	年間	（12回）	11,760円	（新規／継続 NO.　　）

〔備考欄〕

この払込通知票は、機械で使用しますので、下部の欄を汚さないよう特に御注意ください。また、本票を折り曲げたりしないでください。　（郵　政　省）

コンピュータアーキテクチャ編

減算器回路の発展形

Misawa Kazuhiko
三沢 和彦

今月は加減算回路製作の前半部分である，表示部分の製作を行います。ポイントは7セグメントLEDの使い方，そして補数の表現方法です。いままで学んだ内容を再確認して製作に挑みましょう。

前月は加算器回路を改良して，制御信号により加算と減算とを切り替えることのできる回路を設計しました。減算を実行するために，まず負の数を表現する「2の補数」というデータ形式について考えてみました。そして，この補数表現による負の数を使えば，加算器と減算器とはまったく同じ回路でできることがわかりました。

問題なのは，正の数から負の補数表現へ変換する部分で，これはXOR回路を使った反転回路で実現される，ということも2月号で説明済みです。ひととおり設計の終わったところで，今月の課題として残したところは，負の数を出力表示するのに，補数のままだと一見しただけではわかりにくいため，表示を見やすくする回路を加えようという点です。たとえば，皆さんは2進数の1011を見て，即座にそれが−5であることがわかりますか？　今月は，まず1011というデータを入力すると−5を表示してくれるような回路を設計，製作していきたいと思います。

7セグメントLED

最初に基本的なところから始めたいと思います。すなわち，正の2進数データを入力すると10進数の数字を表示してくれるような回路について考えていきましょう。現在多くのコンピュータシステムにおいて，コンピュータの演算結果を表示する出力装置はテレビのブラウン管を使ったCRTディスプレイが主流です。このほか，最近ではノートパソコンで使われている液晶ディスプレイも主力に加わってきました。しかしこの連載で設計製作していくシステムは，CRTや液晶ディスプレイを使うとなると難しすぎて，とても入門記事のレベルに収まるものではありません。

そもそも現在開発中のシステムは，当面は簡単な加減算器なので，数字だけ表示できれば十分です。そこで，数字を表示してくれる回路として，電子式卓上計算器でも使われている，7セグメントディスプレイを取り上げてみたいと思います。

完成した加減算器

7セグメントディスプレイとは図1のような棒状のセグメント(segment：切片，線分などの意）からなる表示回路です。電卓に使われているディスプレイは液晶でできているものですが，今回はLEDでできているものを使います。セグメントがLEDでできているものは，図中に示すように発光するセグメントを組み合わせて，1桁の10進数0～9を表現します。表示したい数値データに対して，実際に発光させるセグメントの対応は図2のようになっています。ビットと表示部分が直接対応しておらず，あまり簡単ではありません。そこで，その対応を変換するためのデコーダという回路

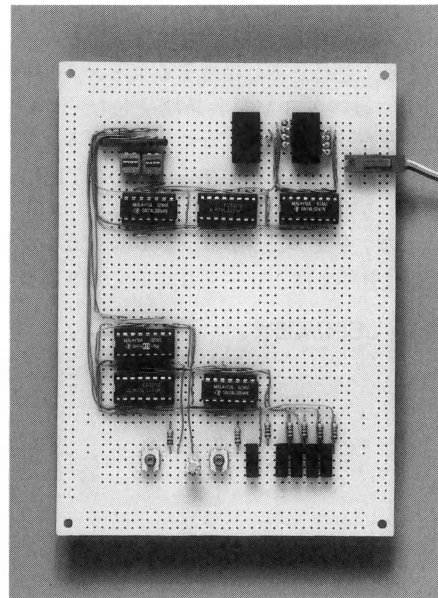



図1　7セグメントディスプレイ

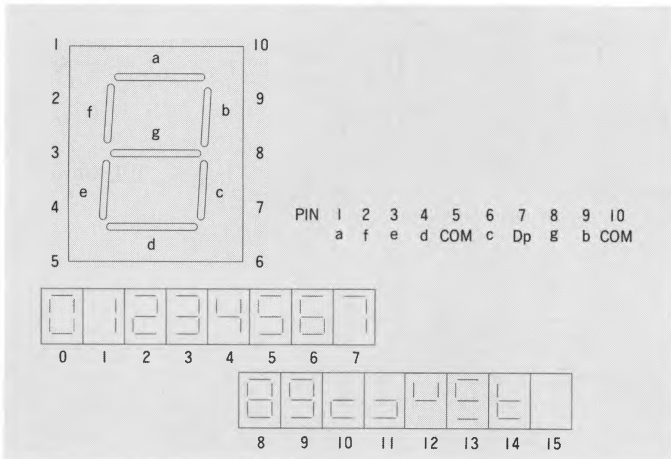



図2　入力データと点灯するセグメントの関係

点灯するセグメント

入力データ	a	b	c	d	e	f	g
0＝0000	○	○	○	○	○	○	×
1＝0001	×	○	○	×	×	×	×
2＝0010	○	○	×	○	○	×	○
3＝0011	○	○	○	○	×	×	○
4＝0100	×	○	○	×	×	○	○
5＝0101	○	×	○	○	×	○	○
6＝0110	○	×	○	○	○	○	○
7＝0111	○	○	○	×	×	×	×
8＝1000	○	○	○	○	○	○	○
9＝1001	○	○	○	○	×	○	○

がTTL ICのパッケージになっています。TTL回路において一般的に使われているICはLS247で，この規格表を抜粋して表1に示します。

LS247は入力に上位ビットからBCDAの順番で4ビットの入力端子があります。4ビット入力なので0〜15まで入力できますが，10以上になると意味のない表示になってしまいます。出力はa〜gの7つのセグメントに対応しており，抵抗を通してLEDに接続すると，入力した2進数に対応する10進数の数字を表示してくれます。出力は負論理で，すべて消灯の状態では出力はすべてHレベルになっており，点灯させたいセグメントの出力がLになるようになっています。そのために接続する7セグメントLEDには，＋5Vが共通端子となっているような「アノード・コモン」というタイプを選択します。今回使用するのはTLR313という型番のものです。

この7セグメントディスプレイデコーダLS247を使えば，0〜9に対応する2進数データをそのまま入力するだけで，自動的に10進数の数字を表示することが可能になるのです。

今回の回路において表示させたいのは，4ビット2進数に対応する10進数で−8〜7となっていますが，少なくとも正の0〜7については，データをそのまま7セグメントデコーダに入力することで表示させることができます。問題は負の数については補数表現をとっているので，そのままデコーダに入力してもまったくでたらめな出力が出てきてしまうことです。そこで，次に補数表現による負のデータを7セグメントLEDを使ってどう表示するかという点について詳しく考えていくことにします。

負のデータ表示回路

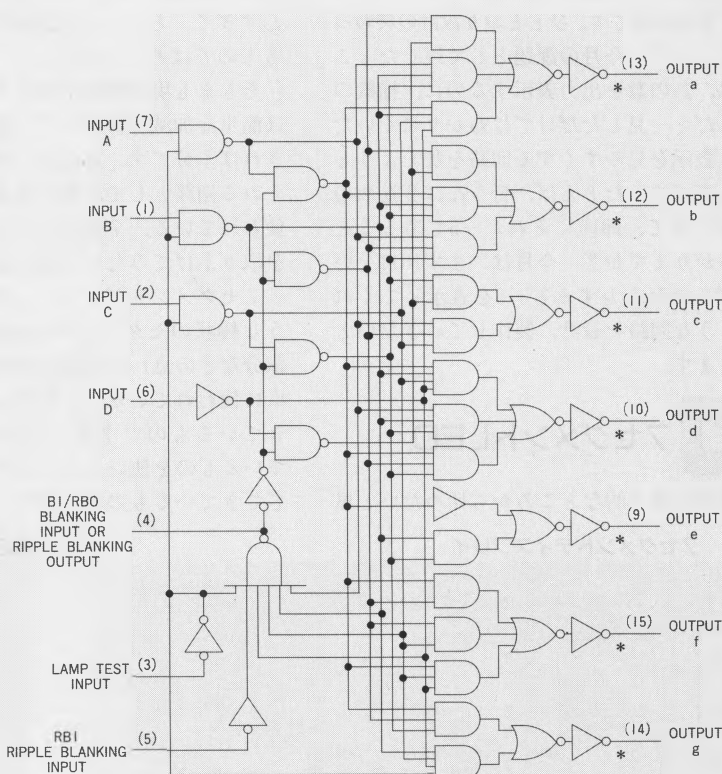
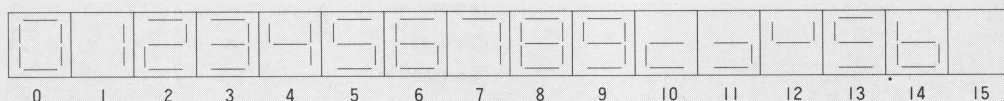
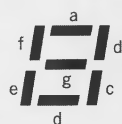
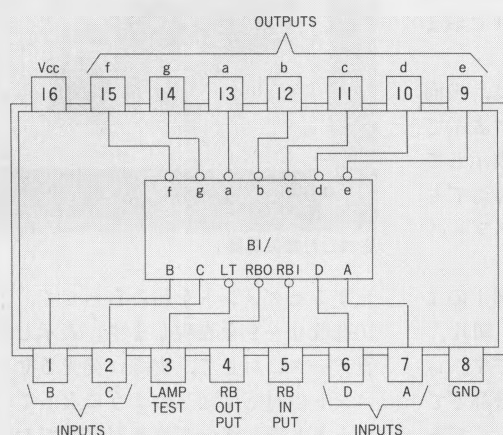
今回設計したい回路の機能は，正の数が入力されたら素通りにし，負の数が入力されたら，その絶対値とマイナス符号とを別にして出力するというものです「絶対値とマイナス符号とを別にして出力する」という意味がわかりにくいと思いますので，具体例を挙げて説明してみます。この回路において，入力は4ビット2進数で，10進数にすると−8〜7に対応します。

0000＝0，0001＝1，0010＝2
0011＝3，0100＝4，0101＝5
0110＝6，0111＝7
1000＝−8，1001＝−7，1010＝−6
1011＝−5，1100＝−4，1101＝−3
1110＝−2，1111＝−1

重要なのは，正の数と負の数との区別は，最上位ビットが0ならば正，1ならば負ということで区別できる点です。

さて，0~7の正の数が入力されたときには，そのまま出力してやりますが，−8〜−1の負の数が入力されたとき，どのように判断していくべきか考えていきましょう。

たとえば−5を意味する1011が入力されたものとします。表示されるべき文字は「−5」で，これはマイナス記号の「−」と絶対値である「5」の2文字からなって

表1　LS247規格表

います。「－」を表示するためには入力データが負の数であることを判断しなければなりません。さらに，7セグメントディスプレイで「5」を表示するには，2進数で5に対応する0101をディスプレイデコーダに入力しなければなりません。

そのためには，

1011＝－5 → 0101＝5

という変換回路が必要となるわけです。

この回路は，2月号で正の数から負の数に変換する回路を設計したときと逆の手順を踏めばよいことになります。しかしながら，実際は先月説明した方法とまったく同じでよいのです。まず，変形前の数値データの各ビットについて，0と1とを反転させます。そして反転後のデータにそれぞれ1を加えるのです。

(反転)(＋1)

1111→0000→0001……(1)

1110→0001→0010……(2)

1101→0010→0011……(3)

1100→0011→0100……(4)

1011→0100→0101……(5)

1010→0101→0110……(6)

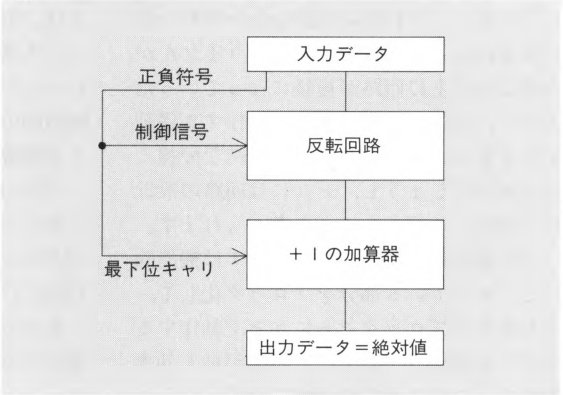
1001→0110→0111……(7)

すると，負の数が正の数に変換されているのです。

ということは，図3のように，反転回路と1の足し込み回路との組み合わせを使うため，先月とまったく同じ回路でよいということになります。

反転回路にLS86，1を足すのにLS283を使うことになります。

ただし，1を足すだけでよいので，下からの繰り上がり端子に制御信号を入れるだけになり，加算器LS283の片方の入力はすべて0にしておく必要があります。そして，正負の切り替えには，入力データの最上位ビットを反転回路の制御信号で実現できるでしょう。これは，入力データの最上位ビットが0のときには正の数なので，素通しにし，最上位ビットが1のときには負の数なので，反転させることになるからです。

負の数を7セグメントLEDで表示するのに，もうひとつ加算器が必要となるのは，なんとももったいない気がします。ところが，よく調べたところ，現在のTTL ICシリーズには正負を変換する手頃なパッケージがなく，自分で加算器を組み合わせるほかないようです。

先月と同じ回路といっても，すべてが同じというわけではないので，実際の回路図(図4)を見ながら，7セグメントLED表示回路の動作をもう少し詳細に追っていきましょう。

まず，入力端子と反転回路のLS86との接続部分に着目すると，入力の下位3ビットがXORゲートの片方の入力になっています。そして，最上位ビットが制御信号として，すべてのビットのXORゲートのもう片方の入力につながっています。減算器では，4ビットすべてを反転させるために4つのXORを使って全ビットを反転させます。さらに制御信号が別にきていたのに対して，こちらの回路ではこの部分が若干違っています。もうひとつのXORゲートは「－」を表示させるために使用していますが，この役割についてはあとで述べます。

加算器のLS283は，先ほど述べたように

図3　絶対値変換回路のブロック図

表2　部品表

IC基板ICB-96PU（サンハヤト製）	1枚	535円
LS86	1個	30円
LS283	1個	60円
LS247	1個	80円
ICソケット14ピン	1個	30円
ICソケット16ピン	2個	@35円
4ビットDIPスイッチ	1個	150円
TLR313	2個	@210円
10kΩ 4素子アレイ抵抗	1本	50円
560Ω 抵抗	8本	@1円
2ピンコネクタ	1個	150円

図4　正負2進数→10進数変換表示回路

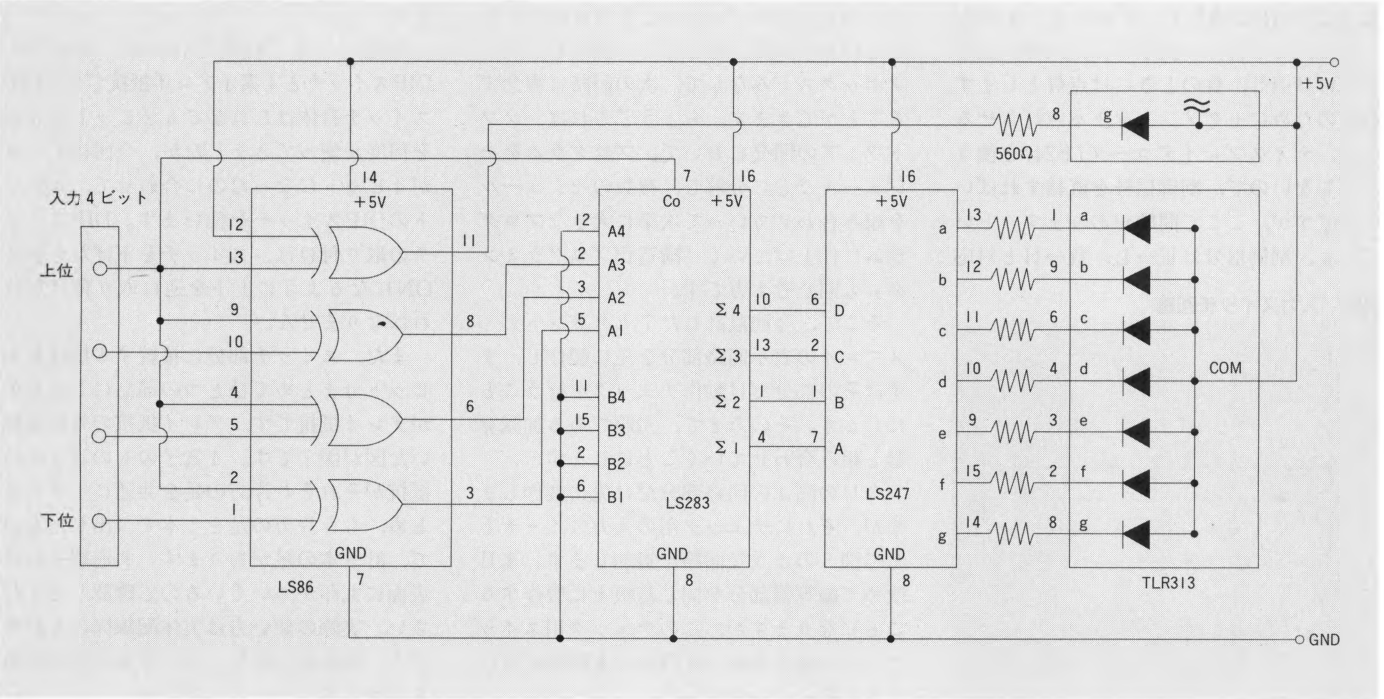



illustration:Y.Kawahara

負の数から正の数への変換時に必要な，1の足し込みを行うだけのために使用しています。1を足し込むだけなので，B入力はすべてGNDに直結しており，これは0を足すことに対応しています。

また，反転回路からの3ビットがA入力の下位3ビットにつながっており，最上位入力はやはりGNDに直結して0の入力になっています。制御信号が最下位キャリ入力につながっているのは，先月の減算器と同じです。LS283の出力が上で述べた7セグメントディスプレイデコーダの入力4ビットに直結されています。

あとの7セグメントLED TLR313の周辺はスタンダードな回路になっており，どんな場合でも同じ回路と考えてよいでしょう。「－」表示のためにもうひとつTLR313を使っています。7セグメントあるうちの1本しか使わないので，もったいないようですが，実際にマイナス符号の雰囲気を出すためにあえて使いました。見た目にこだわらないという人は，ごく普通のLED（たとえばTLR113Aなど）でもかまいません。

さて，この符号の表示には，入力が正のときには消灯，負のときには点灯とします。符号のためにセグメントを1本点灯させるには，ディスプレイデコーダLS247を使うまでもないので，制御信号を直結すればいいのですが，ここで問題があります。というのも，制御信号は正→L，負→Hと対応しているのに対し，LEDのほうはL→点灯(負)，H→消灯(正)と論理が逆転しています。そこで，LEDに入力するためには，制御信号の論理を反転しなければなりません。LS86のゲートで余っている1個はこの論理反転のために使っているのです。これは，XORの片方の入力を常にHにしておけば，もう片方の入力と出力との関係がNOT回路と同じになることを使っています。これによって，符号ビットがL(正)のときに出力H，H(負)のときに出力Lとなって，正しく「－」記号を表示させることができます。

図5　入力スイッチ回路

7セグメントLED表示回路の製作

先月設計した減算器回路と今回設計した回路とを組み合わせれば，目的の回路が完成します。しかし，いきなり全体を一度に製作し始めるのは得策ではありません。というのも，この程度の回路なら全体を一度に作るのもそんなに労力がかかりませんが，今後これ以上に回路が複雑になっていったときに，全体が一度に正常に動作する保証がなくなります。これには，単純な配線ミスもあるでしょうし，ときには回路の設計に一部分ミスがあることも考えられます。

どんなに複雑な回路でも，まずは機能的にまとまっている部分をブロック化して，それぞれのブロックごとにうまく動作するかどうかチェックしていく方法が最も効率的です。このように機能的に分解したそれぞれのブロックを「モジュール」と呼びます。ひとつのモジュールで正常に動作することがわかれば，あとはその回路をブラックボックスとみなして，次の回路に専念することができます。ちょうどこれは，ソフトウェアの開発において，プログラムをモジュールごとに分解し，複数のモジュールを組み合わせていって次第に大きなプログラムに仕上げていく「構造化プログラミング」と同じ考え方です。

そこで，今回設計した7セグメントディスプレイの表示回路部分を先に製作し，まずはその部分だけ動作チェックを行うことにします。そのあとで，主題である加減算器と組み合わせていくことにします。

今月の図4の回路部分だけ先に製作しますが，それにチェック用の入力スイッチとして図5のような回路を追加します。来月改めて減算器部分を同じ基板上に製作することになりますが，このチェック用スイッチはそのまま基板上に残しても問題がないようになっています。

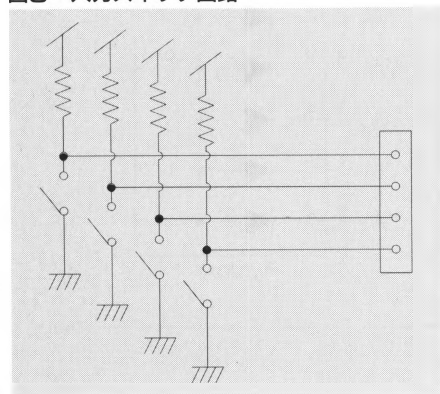
いつものように実体配線図を図6に示します。今回使用したIC用基板はサンハヤト製のICB-96PUというもので，いつも使用しているものよりも数倍も大きいものになっています。これは，今回の表示回路だけでなく，減算器回路とその次に製作する予定の10進入力回路とオーバーフロー回路とをすべて同じ基板上に載せようと思っているからです。

現在のところ，このうちの表示回路と減算器回路の部分までが完成しており，今月掲載した写真の回路を見るとそこまで仕上がっている様子がわかると思います。今後，全体を完成させていくためにも，ぜひ，今月は表示回路のところまでしっかり誤動作なく完成させてください。

配線はほとんどがジャンパ線なので，対応に気をつけて，間違えないように注意深く配線していってください。実体配線図にあるジャンパ線の飛び先を示した番号（たとえば，283-6はLS283の6番ピンへつなぐ，という意味）を確認し，回路図と照らし合わせながら配線していきます。そのとき，回路図の上で配線済みの線に印を付けていくと間違いが少なくなるでしょう。

今回の回路を実際に製作するうえで最も注意すべきなのは，TLR313周りの基板のパターンをカットする点です。この部分を拡大した図が図8です。×のところがパターンをカットする部分で，カッターナイフで溝を切るのがいちばん簡単な方法でしょう。カットしたあとは，そこに取り付ける抵抗器の足を使って，必要なパターンをつなぎます。

部品のうちで目新しいのは，4ビットDIPスイッチと4素子アレイ抵抗です。DIPスイッチ自体はこれまでも2ビットのものを何度か使ってきましたが，今回はデータが4ビットになったのに合わせて，4ビットのDIPスイッチを使います。DIPスイッチの取り付けは，スイッチを下げたときにONになるように上下を逆に取り付けなければなりません。

また，スイッチ部分に接続する抵抗を4ビット分まとめてひとつの部品にしたものがアレイ抵抗です。アレイ抵抗の外形を描いた図が図7です。4素子のものは4本の抵抗がそれぞれ片方の足を共通に1本にまとめ，もう片方の足を1本ずつ出したもので，計5本の足があります。共通端子には表面に丸印がついているので確認してください。実際の使い方は実体配線図のとおりです。市販品には5，6，8素子などがあるようです。

動作チェック

最後に動作チェックです。チェック用入力スイッチは4ビット2進数で，2の補数表現を7セグメントLEDに表示させるように設計されています。上に示した2の補数と正負の10進数の対応表を確認しながら，入力どおりの表示がなされるかをチェックしてください。誤動作の症状としては，

1) LEDがまったく点灯しない

TLR313周りの配線ミス。特にTLR313の共通端子を＋5Vにつないでいない可能性がある。

2) 7セグメントの表示が数字の形にならない

LS247のa～gの出力端子とTLR313のa～g端子の対応が正しくない。

3) 数字は表示されるが，2進数と10進数が対応しない

LS86とLS283，あるいはLS283とLS247の間の4ビットデータの桁の対応が正しくない。LS283のΣ1～Σ4出力やLS247のABCD入力の桁順をチェックする。

4) 正負が正しく判別されない

入力の最上位ビットおよび，制御信号周りの配線ミスを確認する。

今月は切り替え式加減算器の表示部分の製作で終わってしまいました。来月メインの回路部分（写真ではもう出来上がっている部分）を続けて製作して，史上最低のRISCともいうべき最低限の加減算器を完成させたいと思います。まだまだ，X68000に匹敵するコンピュータシステムにはほど遠いのですが，なんとか計算器と呼べるものまでには仕上げるつもりです。

図6　実体配線図

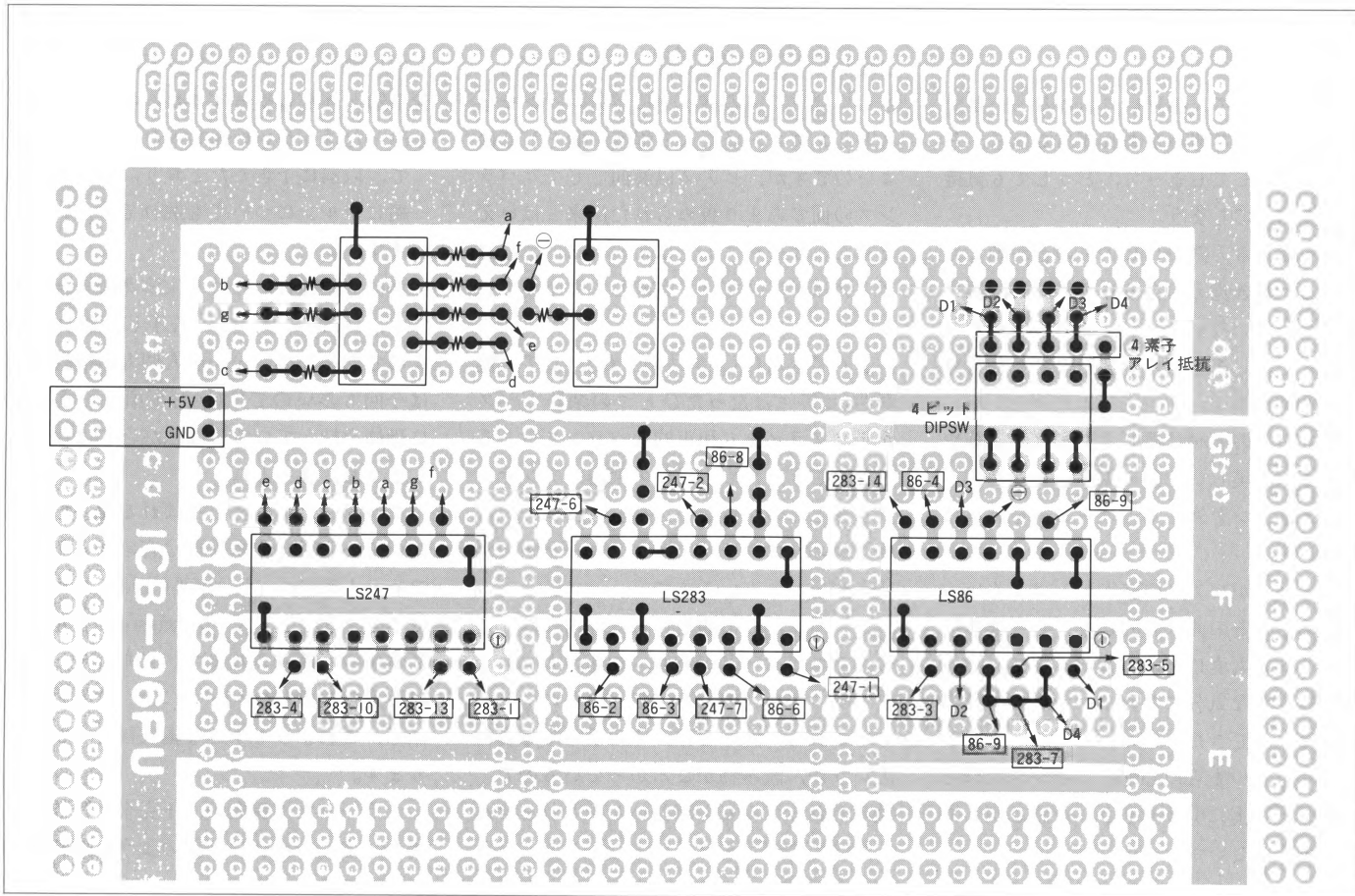
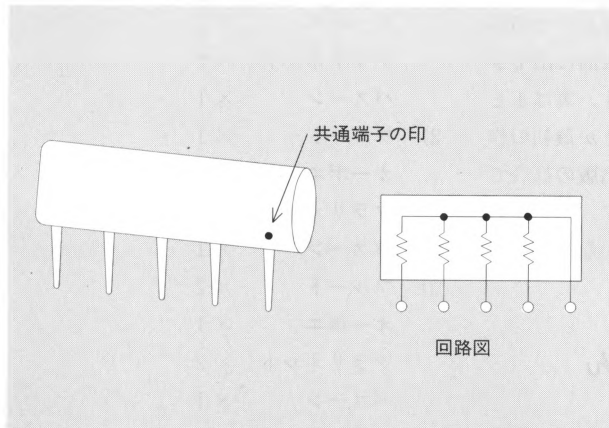
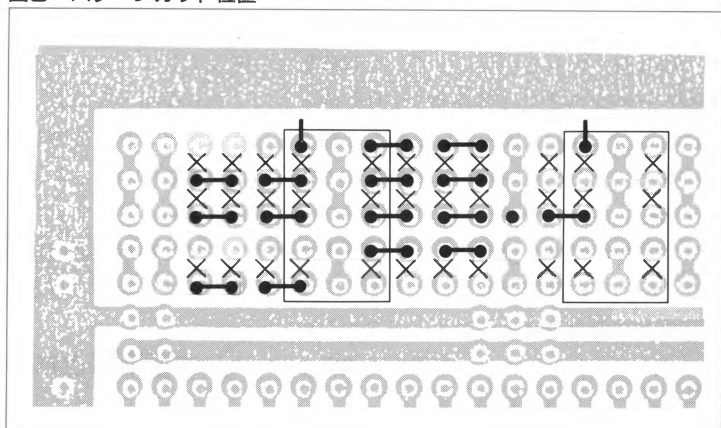



図7　4素子アレイ抵抗

図8　パターンカット位置

CREATIVE COMPUTER MUSIC

Creative Computer Music入門(18)

木管楽器とホルン

楽器の特性についての知識編，今回は金管楽器に続いて木管楽器を取り上げます。同じ管楽器とはいえ，木管と金管では音を出す仕組みが異なるため，音色や演奏するときの特性も違ってきます。ここでは代表的な木管楽器と，音色の似ているホルンを加えた和声の例を紹介しましょう。

Taki Yasushi 瀧　康史

§ピアノの詩人

詩集を開いて，物思いにふける……自分にも同じような想いがあればあるほど，その詩(ことば)に同調(シンクロ)してしまって，どうしても気持ちが高鳴ってしまう。

目を閉じると微かにピアノの音が聴こえてくる。聴き覚えのあるピアノの音。地に足がついていないような，ふわっとした気持ちがして，まどろみを楽しんでいた自分にふと気がつく。

ショパンの調べは優しく，ときには華やかでそして美しい。髪を振りわけながら，ヘッドフォンを外して，コーヒーをいれにいく……。

ショパンがピアノの詩人だとはよくいったもので，その曲は詩的にも，心に染み渡ってくる。素直に感情移入ができて，とっても穏やかな気分になれる。小さじ1杯のシュガーをできたてのコーヒーに，太るかしら？　とつまらぬ心配をしながら入れてしまう。現実に引きもどされた感じがしてちょっと悲しいかな……。

な～んて，ね。ほっほっほ。

今回紹介するのはショパンのピアノ協奏曲です。実はショパンは生涯に2度しかピアノ協奏曲は作っていないので，もしあなたがショパンのピアノ協奏曲のCDを持っていたら，そのCDに入っている曲で全部でしょう(たいていはカップリングされていますね)。

ショパンのピアノ協奏曲を悪くいう人もいますが，それは的を射ているともいえます。確かに曲自身は哀愁があってなかなかなのですが，ピアノと，弦楽器との協奏という部分でいまいちなのです。

たとえば第1番の最初の部分は，まるっきりオーケストラだけで,「これって本当にピアノ協奏曲なの？」といいたくなるような感じです。しばらく聴くとピアノだけ，もしくはピアノのバックにピアニッシモなストリングスが軽く入る程度。曲としてはよいのですが，ピアノ協奏曲としてはバランスの面であまり褒められた出来とはいえません。オーケストラはオーケストラだけ，ピアノはピアノだけで完成してしまっているのです。

よくできた(バランスのとれた)ピアノ協奏曲(ピアノはたったひとつの楽器でバスからソプラノまで和声的なバランスをとることができるため，ほかの楽器とアンサンブルをとるのが難しい)を聴きたいのなら，ラフマニノフやチャイコフスキーを聴いたほうがいいともいえます。

ショパンがこの曲を作った理由には，いろんなエピソードがあり，初恋の人コンスタンツィアへ宛てたものだといわれています。これらに関しては，その手の参考資料というか，読み物がころがっているので，読みあさってみると面白いかもしれません。

余談ですが，先ほどいったとおり，ショパンのピアノコンチェルトは2曲しかありません。どちらも若いときの作品なので，技術的なこだわりよりも，感じたままのみずみずしいショパンの感性が表面に出てきている作品ともいえるでしょう。実は1と2は作成された順序が逆で，2が最初の作品なのですが，これはどうも出版の都合で入れ替わってしまったようです。……どの時代にも編集の都合があるのかな～って。ほんとに余談ですけどね。

§木管楽器とホルンさん

さて今回は，前回の金管楽器に引き続いて，木管楽器に触れようかと思います。ほんとは，楽器のことよりも，アレンジそのものに触れたかったんだけど，弦楽器，金管楽器ときたらやっぱりね。木管もきちんとやったほうがいいんじゃないってことで。

で，以前に予告したとおり，木管楽器と一緒にホルンについても触れてみたいと思います。

前回と同様に，今月も生演奏の場合の条件で，DTMについては若干しか触れませんので念のため。……個人個人の検討事項は今回も多いのです(検討事項という言葉に恐怖を持っているのは私だけではないでしょうね，きっと)。

フルオーケストラと呼ばれる編成では，たいていの場合の木管群は音の高い順に，フルート(flute)，オーボエ(oble)，クラリネット(clarinet)，バスーン(basson)，ともども2本ずつで構成されます。もちろん，これは普通の編成(フルオーケストラ)のときの話で，これよりも小さな編成はたくさんあります。

こういった小編成の，不完全な管弦楽の場合，多くの場合は下記のような編成をとりがちです。

1)	フルート	×1
	オーボエ	×1
	クラリネット	×1
	バスーン	×1
2)	フルート	×1
	オーボエ	×1
	クラリネット	×2
	バスーン	×1
3)	フルート	×2
	オーボエ	×1
	クラリネット	×2
	バスーン	×1
4)	フルート	×1

クラリネット　×1

このようにする理由はいろいろありますが，それは主に楽器の特性に起因するものです。たとえば，どの例でもオーボエとバスーンが1本なのは，オーボエの音はあまりにリーディ(reedy)で，小編成のアンサンブルに2本も使うとまとまりがとれにくくなるということや，バスーンは「パワーのある」という形容があまりにぴったりな音を持った楽器で，これもまた小編成のアンサンブルでは低音域が重くなってしまう，などということが挙げられます。

音の雰囲気の都合上，ホルンもこれらのアンサンブルに含めます。ただし，たとえ小編成だろうと，アンサンブルの都合上，ホルンは1本で使われることは稀なのですが，それについては，アンサンブルのお話のときにでも詳しく説明しましょう。

そんなワケで，小編成ではホルンは少ないときは2本，完全編成のオーケストラでは4本，正確にはツーペアで使われると覚えておくのが的確でしょう。

それでは楽器の個々の詳しい説明から始めましょう。

§フルート

フルートの音域は図1-1に示したとおりです。金管楽器の最高音が演奏者の唇に依存するのに対し，木管楽器はリードの部分で発音するため，最高音から最低音までの音域は誰が演奏しても違いはありません。

では，この楽器を使うときに注意すべき点をいくつか挙げてみることにします。

まず，楽器の特質上，最高音になればなるほど，小さな音で安定して発音させることができにくくなります。若い読者は，小学校もしくは中学校でリコーダーの練習を強制的にさせられたと思うので，そのあたりのニュアンスはわかりますよね。

つまり，まず最高音の2つ，すなわち，BとCは音を出しにくく，微妙な表現は困難なので，*ff*(フォルティッシモ)のみにすべきでしょう。

音質についてもいくつか挙げましょう。

フルートの低音域，図1-2に示す部分は，非常に味わいがあり豊かで，音に潤いがあります。最近のオーケストラの曲で多用されている音といえます。近代ではフランス系の作曲家，ラベルなんかもよく使っていますし，なにしろよく耳につく音です。

この音域でのソロは，非常にはっきりとした音で，豊かなのですが，ほかの楽器に容易に隠されがちなので，うまくバランスをとらなければなりません(それにしても，SC-55だとフルートの味わいがまったくなくって悲しい……)。

また，この音域のフルートは，*pp*(ピアニッシモ)のトランペットの音によく似ています。どの程度似ているかは，ドビュッシーの「牧神の午後への前奏曲」を聴くとわかるでしょう。

図1-3の音域のフルートは，非常に明るい音です。静かな経過句に明瞭な美しさを与えます。ただし，やはりほかの楽器に消されがちなので，ほかのパートは軽めに作っておいたほうが無難でしょう。

図1-4から上の音域では，透明感のあるきらびやかな音質を持っています。目立つ音なので，ほかの木管楽器もしくはヴァイオリンなどのオクターブユニゾンでは，メロディの高音部にかなりの輝きを与えます。

フルートは，図1-2で示した低音域でない限り，アルペジオやレガート，スタッカートなどの表情を非常に軽快につけることのできる楽器です。また，中音域では同一音の1オクターブ違いなどは，非常に速く反復することができます。このこともリコーダーをやったことのある方なら容易に理解できることでしょう。リコーダーの場合の親指ですね。

これらをゲームミュージックなどのアレンジに使う場合，よくある，SSG音源(に近い)のピロピロとした高音域のアルペジオなどが，速度によっては演奏できます。16分音符なら8分音符におとして演奏するといいでしょうし。

また，この連載の読者には，フルートの音を聴いたことがないという方はほとんどいないと思うのですが，音質の特色上，優しくて甘いメロディ，可愛らしいメロディ(某アリ○ソフトのゲームのタイトルの曲のメインメロはフルートとピッコロの絡みがよく似合う。そういえばラ○ス2はMIDI版を作ったような覚えがある……)，そんなのが合っています。ちなみに，以前紹介した「バナナパフェ味のそよ風」の場合のフルートですが，あれは必ずしもフルートが最適ということで使ったわけではありません。ほかによい楽器を思いつかなかったとか，それから，メロディがほかのパートに移行する手法はあの時点ではまだ使っていなかったなどの理由でフルートを選んだということです。純粋にフルートについてだけのお手本という意味では，あのフルートのパートはあまり参考にしないほうがいいでしょう。

最後に，フルートではできないことをお教えしましょう。

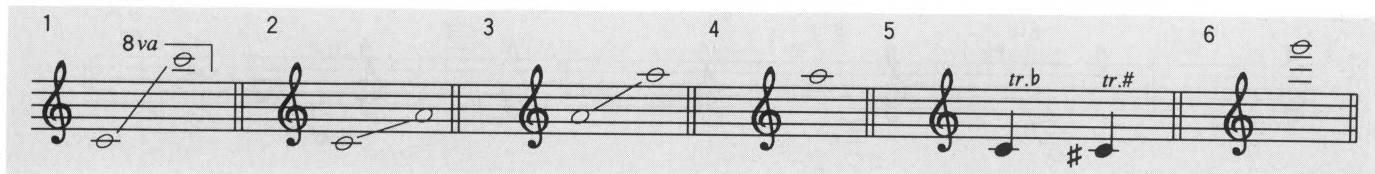
図1-5のトリル，または図1-6の音以上の高さのトリルは，フルートではほとんどできないといってよいでしょう。

§オーボエ

オーボエの音域は図2-1に示したとおりです。この楽器の音は非常に特徴的で，自己主張の強い，いわば鼻声のかかったような音を持つ，ダブルリードの楽器です。

そのため，下手に和声的にアンサンブルをとると非常に浮いてしまい，和声どころではなくなってしまうため，注意深く扱わねばならない楽器だといえるでしょう。小編成の不完全な管弦楽のときに，1本しか使用できないというのはそこに起因します。オーボエは本来，旋律的な楽器で，軽めの音楽における持続音が魅惑的な楽器なのです。

ちなみに，どのくらい自己主張が強いか

図1　フルートの音域など

といいますと，一度オーボエのソロを聴いてしまうと，あまり耳が鍛えられていなくても，小編成のオーケストラならば，下手をしたら最初から最後まで，オーボエの旋律を追うことができてしまうぐらいです。当然のことですが，完全編成のオーケストラでは，オーボエの音の突出を和らげることはできます。

さて，てっとり早く，注意事項を述べておきましょう。図2-2の音より高めの音は，柔らかさがまるでなく，むしろ硬すぎていて，使用するには耐えられない音です。この効果を狙って使うならともかく，単純にこれ以上の音がほしいのならば，フルートや，あとで述べるピッコロを利用するほうが賢い選択といえます。

また，先ほどいったような，旋律的な持続音が魅惑的というのは図2-3に示した音階の範囲です。実はこの効果は図2-3のオクターブのうち，この間のCには適用されないというのがオーケストレーションの定石ともいえるのですが，これも必ずしも一概にはいえず，私は一級のオーボエ奏者がCで魅惑的な音の振動を奏でることができるのを知っていたりします。まあ，生で演奏するつもりがある曲を作るのならば，このようなことができる人はそうたくさんいるものではないので使わないほうが無難ともいえるでしょうね。

オーボエは，その音の特徴から，ソロとして使うには最適な楽器ともいえます。際だった旋律や，表情的な経過句の表現では，オーボエの能力をまざまざとみせつけられてしまいますし，性格がはっきりとした楽器なので，オーボエが短めのパートを吹き，またほかの木管楽器がそれに応えるように吹く，といったような木管楽器同士の対話のようなことにも十分使えます。このときは，図2-4に収まる音階にしたほうがよいでしょう。

オーボエのスタッカートは非常に軽快に行うことができます。速い反復もです。本などによっては禁じているものもあるのですが，低音域の反復も，*mf*（メゾフォルテ）以上にもなると，豊かな音質で奏でると思います(ただし使い方によってはバスーンの音に消されてしまう場合もあります)。

最後に，図2-5以上の高音のトリル，もしくは図2-6のトリルは避けるべきです。

そうそういま思い出したのですが，X68000のユーザーは知らないでしょうか？昔，ファルコムから出ていた「ザナドゥ」というゲームがありましたよね。あのシナリオ2の，通称パチンコ面と呼ばれる行商場所(ここまでいくとほんとに一部の人にしかわからないって……)のBGMのリードをオーボエでやるとかっこいいです。まあ，暇があったらアレンジしてみると，面白いかもしれません。短いので，達成感も得やすいし，曲自体が比較的わかりやすい構成であるという点でも，アレンジの勉強の教材にはいいかもしれません。

§クラリネット

クラリネットは，あの，何だっけ？「ぼ・く・の・た・い・せ・つ・な・く・ら〜り・ねっ・と」っていう曲。あれで結構有名ですよね。

クラリネットには，金管についての説明のなかで怒濤のように出てきた，あの「なになに管」というのがあります。

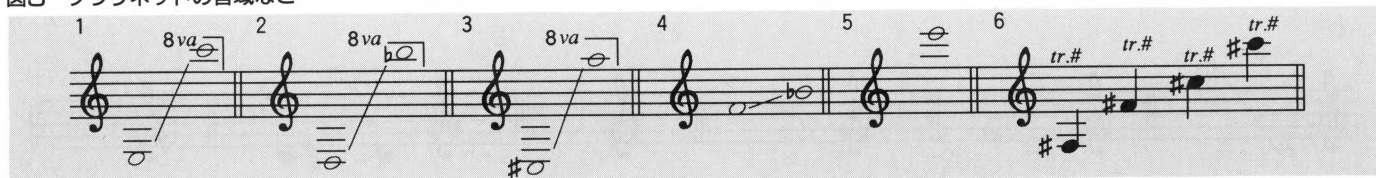
現在よく使われているものは，2種類ありまして，それぞれB♭管とA管です。先月号あたりを読んでいない方のためにもう一度説明しますが，この「なになに」というのは，基音のことです。つまり，Cを吹くと（Cの指遣いで息を吹き込むと），B♭管ならばB♭の音が出て，A管を使うならAの音が出るということなのです。すなわち，楽譜もそれに準じて，B♭管クラリネットの場合，実音より長2度高く，A管クラリネットの場合は原調より短3度高く記譜されます。

クラリネット奏者向けの楽譜を作る場合，これらがなかなかアレンジャーに負担をかけるのですが，まあDTMの場合，コンポーザーが楽譜を読むときに必要となる知識であってそれほど重要ではないでしょう。ただこの連載は，どうもブラスバンド系の人たちにも結構読まれているそうなので(非常に嬉しいのですが)あえていいますと，たとえば原曲がE♭であるなら，B♭管が選ばれてF調で記譜されるということですね。いや〜面倒くさいのですよ，これが。ブラス系かクラリネット奏者でないかぎり(といい切ってしまうのはまずいのですが，話のノリでね)，頭の中ではC調はC調でね〜うんうん。

なぜ，E♭ならばA管ではなくB♭管を使うのかといいますと，♭（フラット）の付く調はB♭管で書き，#（シャープ）の付く調はA管で，というように相場が決まっているからです。しかし，これにもやはり例外はあり，いくらそういう定石があるとはいえ，G♭やC#は，それぞれ遠い調ですから，異名同音のF#，D♭に考え直して，前者にはA管を，後者にはB♭管を用いるのが常です。

当然ながら曲の途中で転調して，その一部だけ原調から遠ざかることはままありま

図2　オーボエの音域など

図3　クラリネットの音域など

す。ただだからといって，クラリネット奏者にB♭管からA管に持ち替えさせるようなことをさせるのはもちろんタブーです。木管楽器は気温が暖かいほど音がよい……というのを聞いたことはありませんか。つまりそれは，管楽器は温度差により周波数が狂いやすいので，管を十分に温めたうえで音の調整をしなければならないということです。曲の途中で，温まっていないピッチの狂った楽器に持ち替えるのは避けたほうがいいでしょう。曲のアレンジの際には，そういうことがないように注意して曲を作るべきです。

それから，A管でもB♭管でも，音の違いはないといっても問題はありません。2つの管は指遣い，技巧的にもまったく同じといっても過言ではありません。なにしろ，これらはすべて調の単純さに基づいて作られているのですから……。

さぁて……いやはや，DTMしかしない人，ご苦労さまでした。

遅くなりましたが，記譜されるときは図3-1のように書き，実音ではB♭管は図3-2，A管は図3-3というのがその音域です。

ちなみに，下は記譜でO3(へ音記号中の)Eから始まっていますが，普通はへ音記号は使われません。

クラリネットの低い音は，「シャルモー音域」として知られていますが，この音は非常に個性的で，特に音が*sf*(スフォルツァンドSforzand：その音だけ強く)で始まるときや，または持続音が増大してから消えさるとき，その音色は異常なほど凄味があり，後ろめたさが漂います(実際に聴いたことのない人には，具体的にどんな音なのかは，絶対わかってもらえないですよね，この表現……。でもしようがない……)。曲のなかでクラリネットを使うなら，この音を覚えておいたほうがいいでしょう。一度は使ってみたい音，でしょうか。

それから，図3-4に示す範囲は，あまりはっきりしない音になりがちです。クラリネットのソロではこの部分の音域は使用するべきではないでしょう。指遣いの困難な部分ともいえるのですが，アルペジオや，経過的な旋律はもちろんできます。ただし，あまり面白みのない音なので，避けたほうが無難ともいえます。

また，クラリネットが高い音を出すと，完全なソロ楽器になりがちです。実際にどの程度までソロに使えるかといいますと，図3-5に示した音まででしょう。人に聞いた話では，抜群にうまいクラリネット奏者はO6Aくらいまでの音を，豊かにしかもピアニッシモで演奏できるそうですが，こんな人はそうそうどこにでもいるわけじゃありませんし，DTMが目的ではなく，クラブなどで学園祭や定期演奏会の発表のためにアレンジをしたくてこの記事を読んでいるような方々の場合は，特にアレンジの際に注意すべきでしょう。

もちろん，図3-5の音より高い音も発音は可能です。ただし，ここを超えたあたりから，奇妙でとてもじゃないがソロでもアンサンブルでも使えないような金切り音になりがちなので，現実問題としては使えないと判断すべきでしょう。

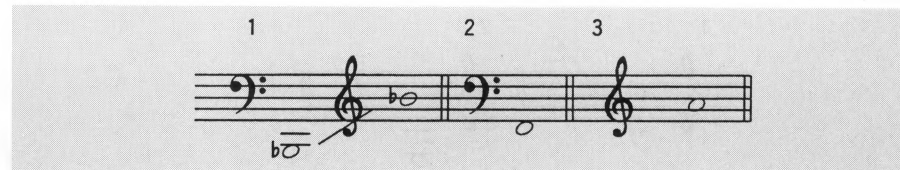
クラリネット自身，軽快で，アルペジオや経過句を得意とするような楽器です。レガートやスタッカートなどを駆使した広い跳躍，速い*p*(ピアノ)と*f*(フォルテ)の交互などのいずれもうまくこなすことのできるような表現力のある楽器ですから，無理してへんてこな音を使うこともないでしょう。それから，クラリネットの3度での重音の経過句，6度での重音の経過句は，もう定番です。一度は使ってみるのがよいかと思います。

最後に，図3-6にあるトリルは避けてください。

§バスーン

別名，ファゴット(fagotto:イタリア名)と呼ばれるこの楽器は，木管楽器の低音部を支える重要な楽器です。

バスーンは移調楽器ではありませんが，3オクターブの音域を持っています。図4-1に示すとおりです。

さらに高い音も可能で，ストラビンスキーなどは「春の祭典」で，最高音の不気味なソロをこのバスーンに持たせています。ただし，普通の吹き手のことを考えると，図4-1あたりで抑えておくのが妥当な線というところでしょうか。

バスーンは一般に低音専門の楽器で，旋律的なフレーズにはあまり使われないのですが，使われるとしたらその高音部でヴァイオリンやほかの管楽器などで，オクターブユニゾンとともに利用される場合が多いといえましょう。

バスーンとフルートが2オクターブ以上離れてユニゾンすると，旋律的な経過句はたいへん爽快に響きます。また，バスーン2本とクラリネット2本は，四声体において，それだけで重厚なバランスを保つ和声を奏でます。さらにバスーン2本とホルン2本を*p*(ピアノ)以下で演奏すると，4本のホルンに近い音を奏でます。

バスーンはその仕組み上，ダブルリードの楽器なので，オーボエといわば「従兄弟」のようなものといえます。そのため，当然オーボエともよく調和します。それから，高めのバスーンと低めのフルートのユニゾンもなかなかの効果をもたらします。

挙げていくときりがないのですが，バスーンはかなり利用価値が高い楽器でしょう。最低のオクターブは非常にリーディであるし，中音域から高音にかけては，かなり美しく重厚に響きます。そのため，小編成の管弦楽では2本も使うと，勢いがありすぎて，浮いてしまうのですが，完全編成の管弦楽ではこの限りではありません。

注意すべき点は，図4-2の音よりも低い音でのトリルは避けること。それから，各オクターブでのD♭，E♭，G♭上のトリルを避け，最高のA(図4-3)の音も避けることです。

§ピッコロ

バスーンまでで，主に使われる4つの楽器の紹介は終わったのですが，さらに，あれば「おいしい」楽器をいくつか紹介してみましょう。

図4　バスーン

ピッコロについて最も簡単に説明すると，図6のような音域を持った，実音はこれより1オクターブ高い小型の楽器です(べつに口から卵を産んで増殖したり，2人に分かれたり1人になったり，神様だったり大魔王だったり，緑色だったり，でんでんむしのように触角みたいなのがついている人じゃないですよ。絶対に)。

形状からわかるように，当然，音は高く，音自体にパワーがないため，管弦楽の力添えが必要で，単独で用いるのにはいまいち不向きな楽器です。

ですから，完全なソロとしてはきわめて稀にしか扱われません。実際には，ほかの木管楽器の音に高音の成分が欲しい場合にオクターブまたは2オクターブ上でユニゾンして使うことがほとんどです。使い方によっては，フルートより軽快で，さっきいったSSGのアルペジオなんかはむしろ，フルートよりもピッコロで演奏したほうが可愛くていいかもしれません。

図5　フルートとピッコロ

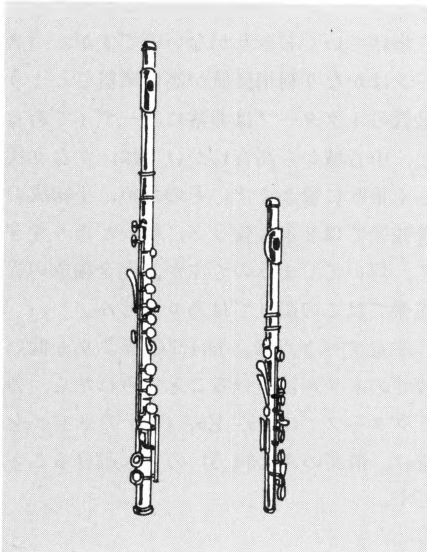
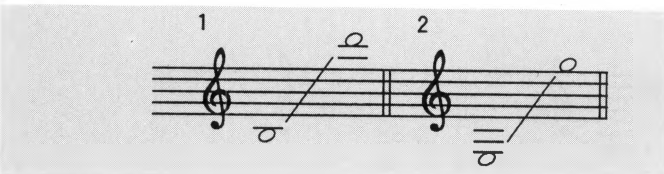



図6　ピッコロの音域

普通は第2フルート奏者が持ち替えて演奏するのですが，大管弦楽では第3フルートが持つ場合がしばしばです。想像すればわかると思いますが，フルートとピッコロの持ち替えにはわずか数秒しかかかりません。

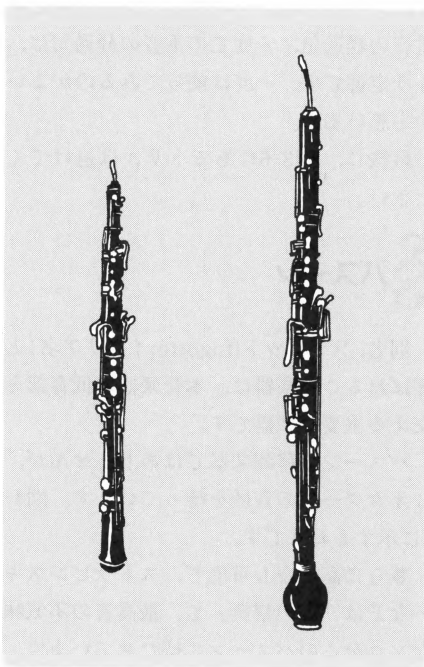
§コル・アングレイ

あれば贅沢な楽器の典型です。簡単にいってしまえばオーボエの大型で，音域は記譜上は図8-1，実音では図8-2で5度低く響く楽器です。

コル・アングレイ，コル・アングレイ，……知らない？　実は，な～んのことはないイングリッシュホルンのことなのです(イングリッシュホルンについては先月号のコラムを参照のこと)。

オーボエの音を知っている方なら音色が想像できるかと思いますが，やはりそのとおりで，オーボエ同様，非常に表情豊かな楽器だといえます。

ピッコロと同じように，コルも第2オーボエ奏者が演奏するのですが，大管弦楽では第3演奏者がします。

図7　オーボエとコル・アングレイ

和声的には，木管の内声を担当する楽器なのですが，やはりオーボエの系統だけあって，ヘマをやらかすと，音が目立ちすぎてしまいます。やはり，せっかくパートに入るのなら独奏楽器でしょう。

重音で使うなら，ヴァイオリン，チェロ，もしくは，低めのクラリネットとほどよく融け合います。

§バスクラリネット

名前から想像できるとおり，この楽器はクラリネットをそのまま大きく，音を低くしたものです。B♭管のクラリネットより1オクターブ低く，管はB♭管だけで，A管はありません。

記譜上では図10-1のように表され，実音では図10-2のとおりです。黒の音符はたいていのバスクラの場合，E♭を持っているということを意味しています。

A管はないといいましたが，実はワーグナーなど何人かの作曲家はバスクラをA管で作曲しています。しかし現在はそれらもB♭管で演奏されています。

やはりコル・アングレイと同じく，これもまた贅沢な楽器です。バスクラリネット

図9　クラリネットとバスクラリネット

図8　コル・アングレイの音域

図10　バスクラリネットの音域

の音は，そのナリからして，確実な低音を管弦楽の上で与えます。これはバスーンを下回るほどです。このバスクラの低めの音は，特質でほかに例のないボリュームのある音を奏でます。この音域でしばしばソロがあるのですが，この音は確かに目立ち，ねらう効果によっては最適でしょう。

チェロ，コントラバスなどとユニゾンをしても曲が映え，これらは効果的な使い方です。しかし，高めの音は個性に欠け，ソロで聴くとつまらないものがありますが，それらは逆に，和声での内声を支える頼りがいのある音になるわけです。

大管弦楽ではたいてい，２本のクラリネットに１本のバスクラリネットとして補充されます。小さめの管弦楽では，第２クラリネットがバスクラと持ち替えることがありますが，バスクラという高価な楽器を持っている人があまりたくさんいないこと，持ち替えを行うには危険なことから，あまりよい手段ではありません。

チャイコフスキーの「くるみ割り人形」の中の「コンペイトウの踊り」に，バスクラが入っているのですが，聴き取ることができるでしょうか。

§ダブルバスーン

ダブルうんちゃら，という名前のものはたいてい，大きくて低い音が出ると思って間違いありません。

そういえば，コントラバスは別名，ダブルベースともいいます。実は「コントラ」というのも「２つの」という意味なのですよね。つまり，ダブルベースとコントラバスはまったく同じ意味なのです。

まあ，ダブルどうのはさておき，名前どおり，このダブルバスーンというのは大型で，低い音が鳴る楽器です。

コンバスと同じく，楽譜は１オクターブ下に書かれます。記譜では図12-1，実音は図12-2のとおりです。

この楽器を軽快に演奏するのにはテクニックを要します。その機能上，低音に厚みを加えることに，存在価値のある楽器なので，コントラバス同様，あまり複雑なパートは与えるべきではありません。

また，ダブルファゴットの音は非常によく通るので，これを抑えるのは非常に難しいため，*mf*（メゾフォルテ）以上という条件のもとで使うべきです。

§ホルン

あえて，木管楽器の仲間のように扱っていますが，ホルンはれっきとした金管楽器です。木管楽器と金管楽器の違いは，実は，木製か金属製かではなく，リードがあれば木管楽器，指遣いによってのみ音が決まれば木管楽器，といったような分け方をします。したがって，サクソフォン（サックス）は木管楽器だし，フルートも銀色のれっきとした金属なのに木管楽器だというわけですよね。

金管楽器であるホルンは，マウスピースと呼ばれる，小さい円錐系のもので唇を押さえつけ，そこで押さえられた部分の唇をうまく震わせて音を鳴らします。

でも，一瞬で音の高さを変えるためのバルブは３つしかありません。単に３つなら組み合わせ上は，６種類……って計算になりますよね。それなのに，音域はそれ以上となっています。

これは，管の長さに依存しているひとつの基音に基づいた，和音の倍音列（この連載第１回目で説明しています。ＣＧ<ＣＥＧB♭<Ｃといったものです）を歌口の変化により変え，唇の振動をコントロールし，音階を出すという……まあ，金管楽器を使った経験のない人間には神業にみえる作業を金管楽器の人はやっているわけです。

当然ながら，いかに美しく音が出るかは，なんといっても「楽器」である人間の「唇」にかかってきます。金管楽器は「共鳴器」にすぎないわけですから，当然，奏者の練習量や素質に大きく依存してしまうのです。話によると，歯並びや唇の形状も美しい音を出す条件のひとつだそうですから，なかなか大変な楽器ですよね。

ホルンも音は金管楽器っぽくない音ですが，当然この条件からはずれてはいません。

クラリネットなどと同様に，ホルンも移調楽器で「なになに管」とかがあります（まあ金管楽器にはたいていありますけれど）。昔のホルン（といってもまだバルブがないころ）はいちおう，全種類があったのですが，いまは使われるのはたいていＦ管です。そのため，記譜では図13-1のように，実音では図13-2のように発音されます。

木管楽器と違って，最高音の限界は指遣いなどでは決まらず，その奏者の唇がいかに小刻みに振動することができるかに左右されます。したがって，ここに示すよりも高い音を出せる人もいるかもしれませんし，逆に一般的な音域とされているなかでも，高いほうの音になると，まともに出せない人もいるわけです。よって実際には，記譜で図13-3，実音では図13-4の程度に抑えておいたほうが無難といえます。

この音より高い音になると，ベテランの奏者なら，*ff*（フォルティッシモ）で，という条件において，鳴らすことができます。これは近代音楽では比較的よく使われているのですが，普通の奏者には逆立ちしても無理だったり，素質のない人には無理であったり（素質のある人しか音楽をやってはいけないわけではないのですから），なにより長期間使うと，奏者，場合によっては聴者にも苦痛を与えかねません。

このような制限はいくつかあります。図13-5の音より高音域で*p*（ピアノ）より小さ

図11　バスーンとダブルバスーン

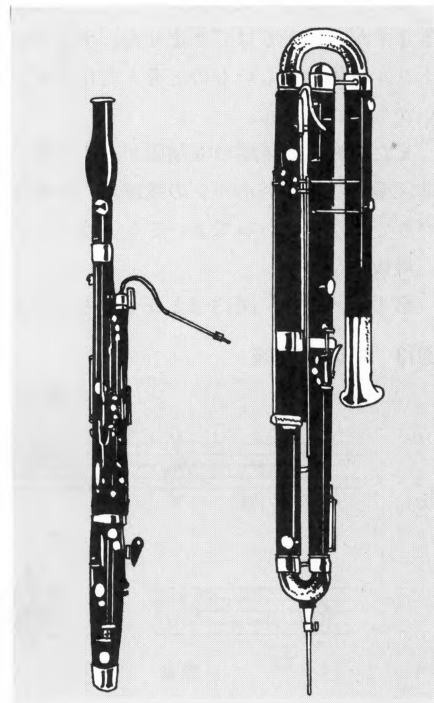
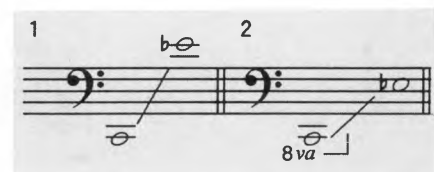



図12　ダブルバスーンの音域

い音を鳴らすことは困難であり，図13-6より低音域では，ただ長い持続音を書く程度に抑えたほうが無難です。

また，図13-7(F調)の中音域において，ホルンは静かで，旋律的な利用をすると表情豊かになります。メンデルスゾーンの「真夏の夜の夢」の夜曲にあるホルンのソロを聴いてみればよいと思います。

強く演奏すると，ホルンの音は拡散し，華やかな音になります。ミュートを付けたホルンで強く演奏すると，荒々しく，甲高い音になります。ミュートを付けた状態で柔らかに演奏したホルンは，非常に表情的で眩惑的です。魅惑的かな？　いや幻想的……いずれにしても言葉ではなかなか伝えられないものですね。

和声的なアンサンブルをとる際に，動かない持続音をとらせることも，ホルンの非常に重要な役目でもあります。しかし，中音域での旋律的なホルンはやはり見事なもので，これをうまく使わないテはないでしょう。またあまりに軽快な反復になってしまうと，音がぼやけがちなので，ほかの楽器と重複させる必要が出てきます。

トリルはある一定の高音部では，利用できますが，ほかではできません。ホルンのトリル自体，珍しいものと考えたほうがいいかもしれません。

また，幅広い跳躍にも制限があるため，非声楽的な幅広いホルンの跳躍は，させるべきではないといってよいでしょう。

最後にひとつ。

第1ホルンは，図13-8より下をとることは稀でなくてはなりません。

第2ホルンは，図13-9より上をとることは稀でなくてはなりません。

どちらもF調ですから念のため。

§ダブテイリング

さて，楽器の紹介をひととおり終えたところで，後編に進みましょう。ここでは，よくよく考えてみれば当たり前のことでも念のため注意点として述べておきます。

楽器を和声的に重複する場合，金管楽器が密集配置，弦楽器が開離配置といった特色があったときと同じく，木管楽器にも和声の置き方にいろいろな制約があります。

ただし，木管楽器はそれぞれの音がまるっきり，似ても似つかないものですから，配置はこれよりもちょっと複雑になってきます。

たとえばここで図14-1の和音を奏でたいとき，オーボエ2本+クラリネット2本の条件のもとで，図14-2と図14-3ではどちらがほどよく融け合うと思いますか？

前に述べたとおり，オーボエはリーディであるし，クラリネットは平滑的です。音色には隔たりがあり，下手に図14-2のような結合をするなら，オーボエが浮いてきてしまいます。

これと同じようなことをうまく行うことが，音色が似ていない木管楽器のハーモニーをうまく成功させる要因だといえるでしょう。ところがこれが，やたらに交互に混ぜればいいというものではなく，たとえば，図14-4のような和声では，第1オーボエが高すぎて，前に述べた使いたくない音域に達してしまうため，もう和声以前の問題になってしまい芳しい効果をもたらしません。また，図14-5では第2フルートが低すぎてしまい，低すぎるフルートは弱くてほかの楽器に容易に消されてしまうため，和声的に音量のバランスがうまくとれなくなってしまうという結末が待っています。

こういうときはそれぞれ，図14-6，14-7に記したように，上部をフルート2本，下をオーボエ2本にしたほうが，よいハーモニーを奏でるというわけです。

こういったダブテイリング(埋め合わせ)を，無思慮に信じてしまい失敗する例はほかにもあります。たとえば，ホルンを結合させるとき，第1ホルンをあまりに高く置きすぎたとき，(高い音は華やかになるのはさっき述べたとおり)その曲の和声がそこで静かなものならば，もう目も当てられないものになるでしょう。

それぞれの楽器が，どのようなときに和声的に融け合う音なのか，どのようなときに特徴的で華やかに響くのか？　最初にこれを説明したのは，これが最優先されるべき絶対的なことで，ダブテイリングにおけるテクニックは二の次の知識だということを忘れてはなりません。

§アンサンブルにおける実例

いくつかの例をもとに，アンサンブルの例を考えていきましょう。

図13　ホルンの音域

図14　和声

まず，図15-1の場合。低めの2オクターブ以上にまたがる広い音域を覆う和音の配置で，気をつけなければならない点は，音を比較的離し，開離配置に近い形で置かなくてはなりません。

ちなみにクラリネットがシャルモー音域ですが，これらは注意が必要かもしれません。

次に図15-2ですが，高めの配置のときは，上部になるほど密集配置をし，下部は開離配置にするべきでしょう。これは，和声的な倍音列が低音域より高音域のほうがずっと密になるためです。

図15-3は，C長調のドミナント7th第1転回形です。全体的に*pp*（ピアニッシモ）で柔らかめの和音をイメージしています。

フルートとオーボエを除くすべての対の楽器はダブテイリングをされています。この下にオクターブ結合をしてバスーンなんかを持ってくると，下部にパワーがあるコードになります。

図15-4は，A短調のトニックです。フォルテで演奏される強い和音をイメージしています。

ここでのポイントは，クラリネットがオーボエの上に置かれている点でしょう。オーボエという楽器は，高音域になればなるほど，音が軽薄になるのですが，クラリネットはまったく反対で，高音域になればなるほど，明瞭で煌きのある音色に変化します。その結果，高音部の和声に強烈なアクセントを与え，*ff*（フォルティッシモ）などのここ一番の部分での印象づけにもってこいともいえるでしょう。

対してホルンは和音の根音を吹き，開離的に置かれた低音部のファゴットと高音部の密集的な和音をうまく結合させるため，高めに共鳴させています。ここでのポイントは，ホルンはほかの楽器に比べて多少弱めに，ほかが*f*（フォルテ）ならばホルンは*mf*（メゾフォルテ）にといった気遣いが必要です。さあ，どうしてでしょうか？　ホルンは高音になるとどうなってくるかは，さっき説明したとおりですよね。

次は，図15-5。これは，C長調のドミナント第1転回形。静かで大きく広がったイメージのもとに配置されています。

低音部から高音部までまんべんなく音が広がるため，この和音には非常に厚みがあります。転回形の低音の条件として，この例のように，透明感のある，響きのよい和音を狙うなら，上部の重複は避けるべきです。もし望むのであれば，下部ではオクターブユニゾンが効果的でしょう。

図15-6は和声的な解決の例。単純なC長調のドミナントモーションです。

ドミナント7thは図15-3のものと同じものです。これをトニックに進行させるのにはそれなりの条件が必要になってきます。

まず，一番のポイントは，「あるひとつの楽器で鳴らされた不協和音は，必ず同じ楽器で解決しなければならない」ということでしょう。これを守らないとどんなことになるかはやってみればわかります。

ただし，これにもある程度例外があります。完全に中絶するか，または休止が不協和音と次の和音を完全に分離して，いわばドミナントモーションとはいえない場合，これに限定されることではなくなります。

§オクターブユニゾンとユニゾン

半ば箇条書きで，定石を覚えているかぎりつらつらと書き連ねていきましょう。どれも音色により限定されたことであり，経験が積まれれば自然に身につくことになると思います(なってくれるといいなぁ～)。

木管楽器のオクターブ結合が非常にドラマチックなのは，おのおのの性格がはっきりしているせいだといえるのですが，以下のものは特に美しく共鳴します。

- フルート，ホルンは，オーボエ，クラリネット，バスーンとオクターブユニゾンすると音に輝きを与えます。
- ホルン，バスーンは，クラリネットとの2オクターブ結合で，羨ましいほど仲のよいカップルになりえます。
- オーボエと，クラリネットのオクターブ結合は，あまり美しくありません。
- オーボエは，バスーンとホルンのオクタ

図15　和音の配置の実例

ーブ結合に味があります。

●クラリネットとバスーンの2オクターブ結合，クラリネット，ホルンの結合は，よい響きを持ちます。

●2つのクラリネットを2オクターブ以上離し，片側，第2クラリネットをシャルモー音域に閉じ込めると，奇っ怪な響きがして効果的です。このとき，中間のオクターブにオーボエを加えると，オーボエのリーディな性格はさらに助長され，たくましい(?)結合をします。

美しいユニゾンというのはいわば半分決まっているようなもので，それらは，中音域より下のフルートとオーボエのユニゾン，フルートと，それにユニゾンできるすべてのクラリネット，低音部のフルートと高音部のバスーン，低音部のフルートとホルンが主なものです。

異色な例をさらにいくつか挙げましょう。

●オーボエをクラリネットに軽く加えると，クラリネットに鋭さを加えた音に聴こえます。

●低音部のオーボエと，高音部のバスーンは，どちらもリーディな従兄弟同士で，異色なフィーリングが漂います。

●オーボエとホルンをユニゾンしても芳しい結果が表れません。ただし，これにクラリネットと，高音部のバスーンのユニゾンを加えると，この限りではありません。

●低音域のクラリネットと，中音域のバスーンは，豊かな結合をします。また，ホルンとクラリネットのユニゾンは，これもまた美しく結合します。

●バスーンと，クラリネットと，ホルンの3本は，魅惑的にホルンを助けます。

ユニゾンは音にボリュームを与えますが，やたらなユニゾンは逆効果になりがちです。もしも，片方を目立たせたいのなら，目立たせたい側を2本にすればよいし，さらに，強めに演奏すればある程度は思ったとおりの性格が出ます。

§悲しいけどまとめ

本当は，もうちょっと，詳しく説明しなければいけないところもあります。新しく曲を作りはしないにしろ，「バナナパフェ～」あたりを木管楽器重視型にアレンジし

木管楽器

先月と同様に，木管楽器の歴史についてちょっと触れてみましょう。話の流れの都合上，一部，先月のコラムや今月の本文と重複があるかもしれませんがご了承ください。

木管楽器と金管楽器の違いは「過去」に木製であったのか金属製だったのかだということは，名前の違いから容易に想像できるでしょう。ただ，実際はこれだけではなく，「演奏法の違い」で分けているのが事実のようです。

金管楽器は本文のホルンの項でも触れていますが，マウスピースと呼ばれる，円錐型の小さい漏斗(じょうご)のようなものを口に押し当てて，唇を振動させて音を発生し，金属の「金管楽器」本体の部分で共鳴させます。

楽器に付いているバルブは，それ自体で音を変えるのではなく，固有の周波数を持つ管に変える仕組みになっています。音程の高低はこの固有周波数を持つある管に，自分の口でその倍音を，いわば「言う」(?)ことで，共鳴をさせて音程を変えます。したがって，たいていの金管楽器にはもともとバルブというものはなく，バルブが発明されるまでは，その管にある基本周波数の「倍音」しか出せませんでした。つまり，たとえばその管がA (440Hz) の共鳴をする楽器であれば，2倍，3倍と，音程にしてみれば，A，E，<Aというようにしか鳴らせなかったのです。巻き貝などから発展したといいますが，その意味もだいたいわかると思います。

いっぽう木管楽器は，まず息を吹き込んだだけで音が鳴るもの……いわゆる「笛」である部分の「リード」を利用して，その小片の笛の鳴らす固有周波数を長さの違う管を通し，音を変えることによって，音階を作っています。

しかし，さまざまな長さの笛をたくさん持ち歩くわけにはいかないので，1本の木の管に穴をあけ，指で塞いだり離したりして息が抜ける長さを調節して音の高低を変えていました。

ここまで考えてみれば，金管楽器と木管楽器の違いはおのずとわかりますよね。

●フルート，ピッコロ

フルートは，木管楽器の代表的な楽器ですが，現在のフルートは金属製です。

フルートの原形ともいうべきものは，古代エジプトや古代ギリシアなど，かなり昔からありました。現在の形に近いフルートは，800年ほど前にアジアから伝えられ，ドイツやスイスなどで民族舞踊や軍楽用に使われていました。管弦楽にフルートが入ってきたのは，それからずっとあとの，いまから200年ほど前で，それに対応するように音色が多彩な表情を持ち，大きな音が出るようになり，それまでのリコーダーの座を奪っていったのでした。

fluteは「みぞ」という意味で，語源はラテン語の「flo」(吹く，鳴り響く)という言葉だといわれています。

現在ではその音域が高音であることや，多彩な表情の音色を持っているから，オーケストラのトップノートを担当することが多くなっています。

ピッコロはフルートの一種で，いまから200年ほど前にフルート以上の高音を求めて作られたものです。大きさもフルートより小さく，管の長さも約半分，音域はそっくりそのまま1オクターブ上なので，フルート奏者がそのままピッコロを吹くことができます。そのため，オーケストラでは第2フルートや第3フルートの人たちが曲の途中でピッコロに持ち替えることもしばしばあります。

piccoloには，「小さい」などの意味があり，楽器名はもともとの名前の「フルートピッコロ」が省略されたものです。そのため，ほかの楽器でも，小型のものをピッコロということがあります。

なお，フルートやピッコロにはリード部分がなく，スレッドに息を微妙な角度から吹き込んで，鳴らすものです。原理はビールビンに息を吹き込んだときと同じようなものですね。

フルート

英語	Flute
イタリア語	Flauto
フランス語	Flûte
ドイツ語	Flöte

ピッコロ

英語	Piccolo
イタリア語	Flauto Piccolo
フランス語	Petite Flûte
ドイツ語	Kleineflöte

●オーボエ

オーボエは，2本のリード(ダブルリード)を持った楽器です。

本文中に，リーディという形容がところどころに出てきますが，この「リーディ」とは「葦笛的」という意味……といっても，「葦笛って何？」という人も多いはず。

オーボエは，楽器本体にストローのような細い管を通して息を吹き込みます。このストローみたいなものが，オーボエの特徴的なリードなのですが，このリードは，乾燥した葦の管を薄く削って作ったもので，これを2枚重ねて管の吹き口に差し込んで吹き，これらを振動させて音を鳴らすのです。

原理は草笛や，ストローで作った笛(知らないかな？)とまったく同じで，雅楽の「ひちりき」やラーメン屋の「チャルメラ」，それから「コル・アングレイ(コーラングレ。イングリッシュホルンのこと)」も同じようにできています。

リーディといっても，言葉だけではわからないでしょうから，リーディな音を出すストロー笛の作り方の図を示してみます。簡単ですから，作ってみてください。

話を戻しますと，オーボエというのは構造上，温度差による周波数の狂いが少ない楽器です。そのため，管弦楽のピッチ合わせに基準音として使われることも，しばしばあるようです。

オーボエという名前の楽器は600年ほど前からあったそうですが，現在のものとはかけ離れたもので，いまのオーボエの原型といえるものが生まれたのは150年ほど前です。

oboeは，フランス語のhautbois(高音木管楽器)からきたそうで，いまの形になる前には，その音量や音色から，野外での農民や軍楽隊に用いられることが多かったそうです。

英語	Oboe (Hautboy)
イタリア語	Oboe
フランス語	Hautbois
ドイツ語	Hoboe

●クラリネット

これもやはり木管楽器の代表的な楽器です。音域が広く，繊細な表現をすることができるいっぽう，管弦楽や吹奏楽でどの楽器にも合ってしまうという特色を持った楽器です。

演奏にあまり不得手な部分がなく，そのため吹奏楽では，管弦楽でいうヴァイオリンの代わりとして使われます。

楽器としては比較的歴史が浅く，300年ほど前に，シャリュモー(葦，リードを意味するラテン語

直して，和声的な音の結合ではなく，旋律的な結合の部分もやりたかったのですが。まあ，誌面の都合などもあり，今回はここで終わりにします。

今回も先月同様，生楽器の演奏についてのアレンジのことが主になり，DTMにおいてどうなるかまでには至らず，人によっては多少物足りなかったかもしれません。「制約ばっかりで，音楽ってもっと自由なものなんじゃないの？」という意見もあるでしょうが，私は最近，芸術のなかでは音楽が最も制限のある芸術だと思っています。ある人が「つまらないオリジナリティにこだわるようなやつは,小説でも書いていろ」と言っていましたが，私はそこまで強くは思わないにしろ，それに近いことはどうしてもあると思うのです。

しかし，DTMだからオクターブ６以上のバスーンを鳴らせたりできるでしょう。

バスーンとして使わないなら，それはそれでかまいません。自分でそのバスーンを数値的に高くした音が，どのような音とうまく結合できるかは，各自の研究の範囲であって，それを成功させたなら，ぜひ，送ってくれると嬉しいところです。

いいかげんな自由は，本当の意味での自由とはいえないのではないでしょうか。規則は規則だ，とまで強調はしませんが，新しい目論見を考えたならそれで結構。いろいろ研究してみるのも悪くないでしょう。私自身，いまその研究の真っ只中ですし。

さて来月の話ですが，気合と根性があれば，旋律的な木管楽器の組み合わせ，弦楽器との絡み，金管楽器との絡みをやってもいいかな～なんて思っていたりします。しかし，思っているだけで，それらを全部やるにはまた相当な分量がありますから，もしかしたら，それはひとまずおいておいて，前々からいっているオープニングやエンディングの作り方とか，そういったことをやるかもしれません。いずれにしても，まだ調査段階なので……。だいたい，思うに，オープニングを作るコツなんて本当にあるのでしょうかね～？　まあ，これも研究しておいて損ではないので，いろいろなジャンルの楽譜を探しまくって(もうすでにバンド譜だけで20冊ぐらい集めてある)，考えをまとめています。それではまた。

のフランス語形)から発達し，150年ほど前に現在の形になりました。

clarinetはラテン語のclarus(明るい，声の高い，有名な)が語源で，クリアーや，クリーニング，クリーナーと語源は同じです。楽器名は，当時トランペットのなかに，クラリーノと呼ばれた音の高いトランペットがあり，それに音色が似ているということから名づけられました。この楽器を管弦楽に用い始めたのがモーツァルトなのです。

英語	Clarinet
イタリア語	Clarinetto
フランス語	Clarinette
ドイツ語	Klarinette

●バスーン

日本ではバスーンともファゴットともいわれているこの楽器は，低音の木管楽器で，管が長く，全長３メートルの管を折り曲げて１本の筒のようにした形をしています。楽器の管の長さは１メートル半ほどもあります。管弦楽では低音部を担当している楽器です。

この楽器の祖先もいろいろ調べてみたのですが，どうやら，わかっていないようです。Bassoonはラテン語basis(基礎，土台)という意味から発展した形のようで，バス，ベースと同じような意味とみていいでしょう。フランス語の「Basson」は，オーボエの「hautbois」に対する意味のようです。

いっぽう，ドイツ語のfagottやイタリア語のfagottoですが，これには「束ねられたもの」というような意味があって，楽器が二つ折りにされたことから命名されているようです。

余談ですが，バスバスーン(コントラファゴット)では，全長３メートルどころではなく，６メートルほどもあります。当然，１回折り曲げただけではどうしようもないので，数回は折り曲げているそうです。楽器の先端に付いているピンのところに紐(ストラップ)をかけて首からつるして吹いています。私は持ったことはないからわからないけれど，重いんでしょうね……やっぱり。

英語	Bassoon
イタリア語	Fagotto
フランス語	Basson
ドイツ語	Fagott

●コル・アングレイ

コル・アングレイ(コーラングレ)は別名，イングリッシュホルンと呼ばれます。２月号でちょっとイングリッシュホルンについて書きました。もしかしたら，イングリッシュホルンのほうがやや一般的かもしれませんが，ここでは名前は英語で統一することにしました。

さて，このコル・アングレイは，オーボエの大型のものです。たいてい，管の先が丸く膨らんでいるのですが，楽器辞典などを見ていると，オーボエにもたまにそんな似たような形のものがあって「？」な気分にさせてくれます。

オーボエの音域より低く，長さも１メートル以上あるため，首からストラップでつるして吹きます。音質は豊かで……といってもわかりにくいでしょうが，簡単に知るには２月号でもいいましたがドヴォルザークの「新世界」を聴くとよいでしょう。第２主題の楽器がそれです。もうちょっとわかりやすくいうと，日本ではキャンプファイアなどで「遠き山に日が落ちて」とかいって親しまれているあのメロディを原曲で奏でる楽器です。

あたたかみがあって豊かでしょ？

さてコル・アングレイは300年ほど前にでき,「狩のオーボエ」といわれてきました。コル・アングレイとはもともとフランス語で，cor(角，コルネットの語源)とanglais(イギリスの)の合成語で，つまりは，イギリスのコルネット……イギリスのホルン……イングリッシュホルンという意味です。

イングリッシュホルンと呼ばれるようになったのは，いまから250年前で，現在の形に落ち着いたのは100年ほど前です。

名称の由来は，イギリスにあった狩猟用ホルンに似ているからではないでしょうか？

英語	Cor Anglais (English Horn)
イタリア語	Corno Inglese
フランス語	Cor Anglais
ドイツ語	English Horn

●サクソフォン

今回はやむなく脱落してしまったサクソフォン，通称サックスですが，この楽器のファンも結構多いのではないかと思います。

吹き口以外は総金属製でぱっと見では，金管楽器と思われがちですが，クラリネットと同じような葦製のリードがついており，クラリネットの変形として，木管楽器に分類されています。

音域が数種類あり，その多くはＳ字型をしています。

この楽器はベルギーの楽器研究家アドルフ・サックスが，弦楽器のヴァイオリン属のような高音から低音までの広い音域を同じような音色でカバーできる木管楽器群を作るため，木管楽器の代名詞といえる，クラリネットを改造して作ったものです。

ベルギー人が150年ほど前に作成した楽器ですが，そのへんに歴史的背景があるのかどうか，ドイツ人には100年ほど前までまるっきり使われず，ビゼーの「アルルの女・第１組曲」をはじめとして，サン・サーンスや，ドビュッシーなど，フランスの作曲家によって徐々に用いられるようになってきました。

そのため管弦楽ではサックスはあまり重要な地位をまだ持ってはいないので，今回の連載の内容ではオミットさせてもらったのです。

今世紀になってから，ジャズでよく使われ始めているので，ジャズについて書くことがあったときにコラムにでもしたいと思います。

もっとも，私はジャズは確かに好きなのですが，ジャズを「心から愛している」わけではないので，内容はやや薄くなってしまうかもしれません。

そういえば，中部大に行ったサックス吹きの私の友人は元気なのだろうか，なんて突然思い出しました……。

英語	saxophone
フランス語	saxophone
ドイツ語	saxophon

ストロー笛の作り方

先から２cmほど，上図のようにつぶしたストローを，線のように切る。そして，息を強く吹き込む。強く吹き込まないと音は鳴らない。ストロー本体が短いほうが楽である。こうやってできたべらべらが，ダブルリードである。リーディでしょ。

サクソフォン

あなたにとって なつかしのゲームは何ざんしょ？…
3大なつかしのゲームをあげろといわれれば…
あたくしめのばあい…
ANOTHER CG WORLD

コナミの
けっきょく南極大冒険！
ホイッ
ビクターのMSXでした

エニックスの
ドアドア！
キャ～
これは98

dB-SOFTの
フラッピー！
エビーラ
これも98

年がばれちゃうよねー
いくつかって？
それはひみつひみつひみつひみつの…
このほかにも インベーダーゲーム、パックマン
ニュートロン…などなど…
なつかしの名作ゲームソフトとして再販されるといいのになー
ばれるよそれじゃ
神サマオネガイ…
古川マミーとしては…

キャラも シンプルで かわいかったし…
へー…
キラリ
テク
マク
マヤ
コン
やっぱ おいまー!
あたしたち だけじゃない― じぶんはど―なのよ―
そーだ そーだ
あたしゃ このままよ なんせ もとが シンプルだから…
ついでに脳ミソも シンプルだけど―
なっとく…
あっ忘れてた・今回のCGデータ・使用ソフトは C-TRACEとMATTER あとは1コマ目をみてね.

PENGUIN INFORMATION CORNER

ペ・ン・ギ・ン・情・報・コ・ー・ナ・ー

NEW PRODUCTS

オフィスワープロ「書院」
WD-4800
シャープ

WD-4800

シャープは，ビジネス文書や企画書などをより効率よく，作成できるようにするための機能を装備した，オフィスワープロ「WD-4800」を発売した。

本機は，グラフ作成や住所管理機能，書院カルクなどを備えたオフィス向けワープロで，今回新しく，イラストや飾り罫を加えた文書を簡単に作成できる「アート倶楽部」，定型文書の修正箇所をマークできる「文書活用機能」，ワープロとの対話形式で実用的な文書を作成できる「自動文書作成機能」が加えられた。

これらのアプリケーションは，42Mバイトの内蔵ハードディスクにインストールされており，処理の高速化，操作性の向上を実現している。

辞書は52万語と33万例のAI用例をもつ「A2I-V3辞書」を搭載。400dpiのA3レーザープリンタ「WD-06LP」にも接続可能である。

価格は556,000円（税別）。

〈問い合わせ先〉

シャープ㈱　☎06(621)1221,043(299)8210

PV-F1用カード&周辺機器
PV-1C01/95/CE-PR1
シャープ

CE-PR1

PV-1C01/95

シャープはハイパー電子マネージメント手帳「PV-F1」用の周辺機器として，表作成/計算カード「PV-1C01」，BASICカード「PV-1C95」，ハンディプリンタ「CE-PR1」を発売する。

●表作成/計算カード「PV-1C01」

本カードは，ペンタッチオペレーションで快適な操作ができる，グラフ描画機能を備えた表作成/計算カードである。

予算実績表や交通費精算表など，56種類の定型フォームを内蔵しており，初めてカードを手にした人でも簡単に使うことができるようにもなっている。

従来の「DB-Z」用表計算カード「PA-9C1/2」とデータ互換であり，パソコンソフト「Lotus1-2-3」のデータも「Sheet Link1-2-3」（ハルコーポレーション）を使用することで，データの交換が可能となっている。

価格は22,000円（税別）である（現在発売中）。

●BASICカード「PV-1C95」

「PV-1C95」は「PV-F1」の大画面，タッチアクセス，手書き文字認識の快適な操作環境をフルに生かした，ソフトウェア作成のできる高速漢字BASICを搭載したものである。

本カードを使うことにより，入力ボードとして手書き文字認識や10キーなどを呼び出したり，メニューキーで呼び出すことのできるオリジナルメニューも，簡単に作成可能となる。また，同時発売予定のアイコン作成ツールを使えば，パソコン上でオリジナルアイコンを簡単にデザインすることもできる。

カードの記憶容量別に256K，640K，1Mバイトの3タイプが用意されている。

発売は4月中旬で価格は未定。

●ハンディプリンタ「CE-PR1」

本機は「PV-F1」「PA-V1」「PA-9500/50/9600/9700」で使用可能な，24ドットのラインサーマルプリンタである。

印字最大速度は，約4行/秒（100ドット/秒）であり，フル充電時には約2時間の連続印字ができる。

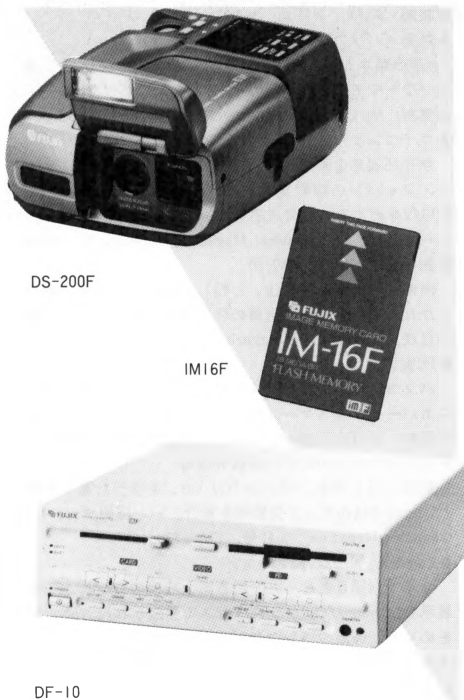
また，「CE-PR1」を使用するために「PA-V1」以外の機種では，別売のBASICカードが必要となる。発売は3月中旬で，価格は95,000円（税別）である。

〈問い合わせ先〉

シャープ㈱　☎06(621)1221,043(299)8210

デジタルカードカメラ/イメージファイル
DS-200F/IM-16F/DF-10
富士写真フイルム

富士写真フイルムでは，デジタルカードカメラ「DS-200F」とイメージメモリカード「IM-16F」を発売した。

「DS-200F」は，39万画素の1/2インチFIT CCDを使用したオートフォーカスデジタルカードカメラである。ビデオ出力による

DS-200F

IM16F

DF-10

再生機能を内蔵しているため，モニタを用意すれば撮影したその場で見ることが可能となっている。

「IM-16F」は，フラッシュメモリを搭載した16Mビットの容量をもった，デジタルカードカメラで撮影した画像データを記録するためのカードである。撮影枚数は，FINEモードで10枚，NORMALモードで20枚，ECONOMYモードなら40枚まで記憶できる。

「DS-200F」の価格は220,000円(税別),「IM-16F」の価格は65,000円(税別)である。

また，同社ではデジタル静止画像を3.5インチのフロッピーディスクに記録，ファイリングできるデジタルイメージファイル「DF-10」を開発した。

「DF-10」は，ビデオから入力された画像や，イメージメモリカードに記録された画像データをデジタル静止画像として，フロッピーディスクに記録することができる。ビデオ入出力にはS端子とBNC端子の2系統が用意され，画像の入力，記録された画像の確認を簡単に行える。

また，フロッピーディスクとメモリイメージカードのどちらへも画像を記録することができ，相互のコピーも行えるので画像データの編集も手軽にできる。

使用できる記憶媒体と記憶枚数は以下のとおり。

・イメージメモリカード
　IM-16F　40枚
　IM-8S　21枚
・フロッピーディスク
　2ED　56枚
　2HD　28枚
　2DD　14枚

発売は4月頃，価格は未定である。

〈問い合わせ先〉
富士写真フイルム㈱　☎0120(209)302

2400bps携帯型FAXモデム
MC24FA5-P
マイクロコア

MC24FA5-P

マイクロコアは携帯型FAXモデム「MC24FA5-P」を発売した。

「MC24FA5-P」は,データモデムとして最大2400bps,FAXモデムとしては最大9600bpsの転送速度をもつFAXモデムである。

データ圧縮機能として，MNPクラス5，CCITT V.42bis,移動体通信対応のMNPクラス10，国際標準規格のV.42（LAPM，MNPクラス4）を搭載している。

外形寸法は，118(幅)×25(高さ)×91(奥行)と小型で簡単に携帯できる。

価格は34,800円（税別）。

〈問い合わせ先〉
㈱マイクロコア　☎03(3448)0811

OS-9でISDNをサポート
OS-9 ISM PortPak V1.0
マイクロウェアシステムズ

マイクロウェアシステムズでは，OS-9でISDNをサポートするためのパッケージ「OS-9 ISM（ISDN File Manager）PortPak V1.0」を発売した。

本パッケージは，68XXXベースのOS-9システムにISDN通信ソフトをインストールするために必要な全ソースコードとオブジェクトコードを含む移植用のパッケージである。ISDN基本インタフェイスをサポートし，ISDNファイルマネージャ，コンフィグレーションモジュール，プロトコルモジュール（ISNネット64，5ESS/5E6に対応），ISMデバイスドライバなどが供給される。

また，AM79C90チップ対応のサンプルドライバも供給され,INSネット64(NTT)仕様の回線交換，およびパケット交換のX.25レベル2までサポートしている。

価格は600,000円（税別）。

〈問い合わせ先〉
マイクロウェアシステムズ㈱
☎03(3257)9000

INFORMATION

パソコン雑誌のオンラインデータベース
パソコン雑誌リファレンスガイド
日能総研マーケティング・データ・バンク

日能総研マーケティング・データ・バンクでは，現在のコンピュータ業界の情報を的確に入手することを目的として，パソコン雑誌15誌の全記事を索引化し，オンラインデータベースによる提供を行う「パソコン雑誌リファレンスガイド」を開始した。

情報源となる雑誌は，PC WEEK，日経パソコン，Oh!PC，ASAHIパソコン，EYECON，THE 1·2·3 MAGAZINE，日経バイト，Macworld，98magazine，月刊パソコン通信，MacJapan，The BASIC，月刊ASCII，MAC POWER，MAC LIFE（順不同）の15誌。

以上の雑誌の記事内容を，記事内容キーワード，フリータイム，雑誌名，発行日，企業名，人名，製品名（ソフト，ハード，周辺機器など），ジャンルコード，記事扱いコードなどの組み合わせで検索することができる。

1991年7月～最新号までデータが収録されており，月に2回データの更新が行われる。

利用料金として，G-Searchでは接続基本料300円/回，出力単価70円/件，NIFTY-Serveでは接続基本料10円/分，出力単価150円/件となっている。

〈問い合わせ先〉
㈱日本能率協会総合研究所マーケティング・データ・バンク　☎03(3432)6970

FILES

Oh!X

このインデックスは，タイトル，注記——著者名，誌名，月号，ページで構成されています。春はもうすぐだけど，まだまだ寒い日も。学生や受験生のみなさん，試験は終わりましたか？　春一番がうれしい便りを運んでくれるといいね。

参考文献
I/O　工学社
ASCII　アスキー
コンプティーク　角川書店
C Magazine　ソフトバンク
テクノポリス　徳間書店
電撃王　主婦の友社
POPCOM　小学館
マイコンBASIC Magazine　電波新聞社
My Computer Magazine　電波新聞社
LOGIN　アスキー

一般

▶'93ゲームミュージックナウ・ゲームメーカーサウンド大列伝
　有名メーカーの音楽担当へのインタビューなど。——編集部，コンプティーク，2月号，別冊付録。

▶ど〜するど〜なる!? パソコンゲーム！
「ついに誕生！　コンピュータソフトウェア倫理機構」。1992年12月からが行されたコンピュータ倫理機構。わいせつ性の高いソフトウェアに対して正式に倫理審査が行われる。そこまでの道のりと読者の反応をレポート。——編集部，テクノポリス，2月号，86-87pp.

▶新鮮良品館
　シャープのワープロ「WD-A751」など，各社の年末年始にかけての家電新製品を紹介。——編集部，POPCOM，2月号，152-153pp.

▶THE NEWS FILE
　ついに10万円を切ったCD-Iプレーヤーや，AT&Tの最新マイクロプロセッサ「ホビット」など，パソコン関連の最新情報。——編集部，LOGIN，1・2号，42-49pp.

▶新製品　Flash NEWS
　シャープのハイパー電子手帳の新ラインナップなど，各社のパソコン，周辺機器の新製品を紹介。——編集部，マイコンBASIC Magazine，2月号，71-75pp.

▶パソコン・キャプテンを使ってみたい人のために
「ウインタースポーツを満喫するお手伝い」。キャプテンシステムをパソコンで有効利用しよう。——編集部，マイコンBASIC Magazine，2月号，76-77pp.

▶ワープロ/パソコン通信新聞
　PC-VANのオンライン麻雀ゲームや大手ネットの各種占いサービスなど，パソコン通信周辺の話題。初心者向け短期連載「パソコン通信への道」は「通信制御手順」のまとめで，プロトコルについて解説。——山本まさこ，マイコンBASIC Magazine，2月号，78-82pp.

▶Bug太郎のプログラム・タイム
「物体変化アニメーションに挑戦」。映画などでおなじみのモーフィング技術のようなアニメーションを簡単なサンプルプログラムとともに解説。——谷　裕紀彦，マイコンBASIC Magazine，2月号，88-89pp.

▶BASICプログラミング講座
「方程式をわかりやすいグラフィック表示にしてみよう」。1次関数や2次関数など簡単な方程式をパソコン画面にグラフ表示するプログラムを作成する。——東　幸太，マイコンBASIC Magazine，2月号，90-93pp.

▶CONTINUE OR NEW GAME
　パソコンゲームの最新情報の特集。X68000用「オーバーテイク」「ストライダー飛竜」を紹介。ゲームの動向を探る。——編集部，ASCII，2月号，181-204pp.

▶革命的なマルチメディアなど存在しない
　マルチメディア改革の筆頭に立つVoyager社のRobert Stein氏と荻野正昭氏に，マルチメディアの今後について聞く。——福富忠和，ASCII，2月号，222-228pp.

▶Digi-Ana Valley
　中央電気の高級CDプレーヤー「TL-1」の仕組みについて聞く。ベルトドライブというアナログプレーヤーの技術でCDの音がよくなるのはなぜ？　——編集部，ASCII，2月号，253-260pp.

▶PRODUCTS SHOWCASE
　松下電器のフロプティカルディスクドライブや，緑電子のニューデザインのHDD「MARiNE」などを評価。——編集部，ASCII，2月号，268-276pp.

▶ことば遊び・コンピュータ
　12月号に続き，ELIZA型会話プログラムを紹介。実際の手法と人工無能との違いについて触れる。——ホーテンス・S・エンドウ，ASCII，2月号，285-288pp.

▶バカパパのモノを買い物
　リラクゼーション関連グッズの巻。手でニギニギ，足でグリグリするものなど。ほかに，テープ起こしや編集に最適なオリベッティのハードディスクレコーダ「QUADERNO」。——バカパパ，ASCII，2月号，336-339pp.

▶ラッキー！ハッピー！オッケー！
　パソコンにまつわる法律上の問題を弁護士に聞く。今月は使用許諾契約の1ソフト1システムについて。——編集部，ASCII，2月号，360p.

▶特集　パソコン通信を楽しもう！
　大手商業ネットの運営の現状や，草の根BBSの紹介。通信ソフトやモデムの購入ガイド，通信ノウハウなど。——編集部，My Computer Magazine，2月号，52-91pp.

▶マイコンからMy Computerへ
　創刊15周年企画最終回。DOS/Vを取り上げ，ASTリサーチ・ジャパンの津村重人氏にインタビュー。あわせてDOS/Vをめぐる市場の反応をパソコンショップに聞く。——編集部，My Computer Magazine，2月号，92-107pp.

▶無手勝流パソコン教習所
「激得！　秋葉原攻略法」と題し，秋葉原での買い物のしかたをレクチャー。店員の存在理由を考える。——島川言成，My Computer Magazine，2月号，152-153pp.

▶PC実験室
　パソコン周辺機器をテストするコーナー。今回はマウスカバーとマウスケースの使い心地をレビュー。——石川至知，My Computer Magazine，2月号，154-158pp.

▶ビジネスマンのための情報管理術
　前回に引き続き，HAL-CATCH Ver.2を使った電子手帳とLotus1-2-3のデータ交換例を紹介。——塚田洋一，My Computer Magazine，2月号，170-173pp.

▶Comdex/Fall'92視察記
　今年で14回目となった，世界最大規模のコンピュータ展示会，Comdex/Fallがラスベガスで開催された。Windowsをめぐる新製品やペンコンピュータの活況，マルチメディアへの試みなどをレポート。——高橋三雄，My Computer Magazine，2月号，203-210pp.

▶ここから始まるスキー場システム
　今年でオープン3年目のスキー場「GALA湯沢」。ここには日本一の規模を誇るコンピュータシステムが敷設されている。その内容をレポート。——大窪志保，My Computer Magazine，2月号，218-221pp.

▶MYCOM WATCHING
　飛行機に関するさまざまな展示がある航空科学博物館。そこで操作できるフライトシミュレータ，マイクロソフトの「Flight Simulator」を紹介。——菊地秀一，My Computer Magazine，2月号，228-231pp.

▶クアンタム社訪問記
　アメリカのハードディスクメーカー，クァンタム社の工場訪問記。そこではハードディスク組み立て体験などもできる。会社の生い立ちや今後の展望も紹介。——編集部，My Computer Magazine，2月号，232-233pp.

▶なんでもQ＆A
　書院パソコンのワープロモードで使う「書院カルク」の表やデータをLotus1-2-3に移動する方法など。——シャープ，My Computer Magazine，2月号，262-263pp.

▶特集　DOS/V購入ガイド
　新製品が次々発売されているDOS/V。その登場までの歴史や仕組み，AVやインタフェイス規格などを解説。購入の手引きに。——編集部，I/O，2月号，31-49pp.

▶各種フラッシュメモリ
　次世代の大容量メモリとして注目されるフラッシュメモリ。日立が開発した新方式のフラッシュメモリ・セルの内容を解説。——編集部，I/O，2月号，124-125pp.

▶スーパーコンピューティング入門
　自然の規則性と乱れ方について新しい地平を開いた「カオス」と「フラクタル」の概念を解説。それらの発展を支えてきたスーパーコンピューティングの役割について考える。——林智雄，I/O，2月号，146-149pp.

MZシリーズ

MZ-2500(BASIC-M25)

▶プッシャー＆プラー
　押したり引いたり倉庫の整理。倉庫番風パズルゲーム。コンストラクション機能付き。——謎のパズル大好きおじさん，マイコンBASIC Magazine，2月号，113-115pp.

X1/turbo/Z

X1シリーズ

▶SQUARE
　対戦もできるテトリス風アクションパズル。——森

大典，マイコンBASIC Magazine，２月号，134-136pp.

X1turboシリーズ

▶イタリアンフィーバー「アラソーイ」

皿を飛ばして敵を倒せ！　単純明快２人用対戦ゲーム。――松原拓也，マイコンBASIC Magazine，２月号，137-138pp.

X68000

▶SOFT EXPRESS

名作シューティング「究極タイガー」，プレイヤーが魔王となって戦う「キングス・ダンジョン」，カラフルな正統派パチンコゲーム「パチンコワールド」。機種別ニューソフトインデックスも。――編集部，コンプティーク，２月号，47-53pp.

▶オリンポスへの道

ポピュラスII徹底攻略ガイド。読むだけでみるみる攻略！　３日で999面（がんばれば）だそうだ。――編集部，コンプティーク，２月号，別冊付録。

▶HOW TO WIN

懐かしのゲーム復活！「テラクレスタ/ムーンクレスタ」を紹介。合体ゲームの元祖か？――編集部，コンプティーク，２月号，130-131pp.

▶GAMING WORLD

正義の手から迷宮を守れ!?　立場逆転負の発想「キングス・ダンジョン」を紹介。――編集部，テクノポリス，２月号，23p.

▶爆発　攻略野郎

「信長の野望　覇王伝」を攻略。織田家，武田家，上杉家で1551年のシナリオをプレイしてみよう。――編集部，テクノポリス，２月号，58-61pp.

▶ゲームの達人

F1ゲーム「オーバーテイク」。格闘アクション「ストライダー飛竜」の攻略は中級・上級編。――編集部，POPCOM，２月号，98-101pp.

▶最新ゲーム徹底解剖!!

「三國志III」を攻略。初心に戻って基本から攻めてみよう。初めて「三國志」をプレイする人向き。バリバリにF1をぶっとばせる「オーバーテイク」はエンジンメーカー別一覧表。――編集部，LOGIN，１･２号，182-185，206-207pp.

▶X68000新聞

最新ゲームの「パイプドリーム」「ドラゴンスレイヤー英雄伝説」。ほかに「Ｃ言語講座第８回」など。――編集部，LOGIN，１･２号，304-307pp.

▶CAMEL BACK

波うつ地面を利用してボールを転がす。マウスを使ったタイムアタックゲーム。――加藤淳一，マイコンBASIC Magazine，２月号，139-140pp.

▶トレーニング・ファイター

かわいいキャラのストIIもどき２人用ゲーム。――鈴木健二，マイコンBASIC Magazine，２月号，141-143pp.

▶マリ夫の冒険パート５

大人気のマリ夫を使ったジャンプ・アクション。各ステージのカギを全部取って扉へ向かう。――高橋秀之，マイコンBASIC Magazine，２月号，144-146pp.

▶SFC版　STREET FIGHTERII　～エンディングテーマ～

スーパーファミコン版ストリートファイターIIのゲームミュージックプログラム。要NAGDRV。――荒木潤，マイコンBASIC Magazine，２月号，158-159pp.

▶GAME PARADISE

新作ゲームの紹介。「エトワールプリンセス」「究極タイガー」「ストライクレンジ」「KU² フロントロー」「パチンコワールド」など。ほかに機種別新作一覧表も。――編集部，電撃王，２月号，103-111pp.

▶FREE SOFTWARE INDEX

ここ数カ月間に主要ネットにアップロードされたソフトを紹介。X68000用真剣白羽取りPro68K，JPEG.X，X68.Xなど。――編集部，ASCII，２月号，373-379pp.

▶なんでもＱ＆Ａ

Communication SX-68Kでサポートしているプロトコルの種類は何か，自動ログインプログラムの作成方法は，などの質問に回答。――シャープAVCシステム事業推進室，My Computer Magazine，２月号，260-261pp.

▶HOBBY EXPRESS

天地無用・足場不安のアクションゲーム「ストライダー飛竜」，ビデオゲーム・アンソロジーの第１弾「テラクレスタ/ムーンクレスタ」のゲームレビュー。――あゆさわかつみほか，My Computer Magazine，305-323pp.

▶満漢プログラム全席

ハードコピーツール「HCOPY君」。乗野方式のハードコピーが可能。――坊農誠，My Computer Magazine，２月号，324-333pp.

▶EL･DRV

X68000用音源ドライバ。低機能だがコンパクトさがウリ。――LPZ，I/O，２月号，69-72pp.

▶GCCで学ぶX68ゲームプログラミング

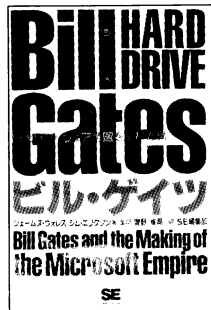
X68000のスプライト機能のひとつであるBG画面をGCCを使って制御する。今回はBG画面を使うサンプルの制作まで。――吉野智興，C Magazine，２月号，140-144pp.

ポケコン

PC-E500

▶INTRODUCE YOURSELF！

戦士の名前をステータスに変換し，戦わせる。バーコードバトラーみたいなゲーム。――宮脇豊重，マイコンBASIC Magazine，２月号，148-149pp.

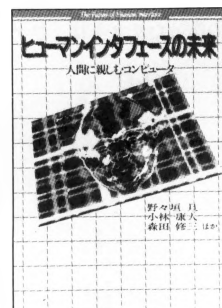
新刊書案内

ビル・ゲイツ
ジェームズ・ウォレス/ジム・エリクソン著
奥野卓司監訳
SE編集部訳
翔泳社刊
☎03(5467)0361
四六判　582ページ
2,800円(税込)

ビル・ゲイツといえば知る人ぞ知る世界で一番の金持ちであって独身であっておたくであるわけだが，彼はいかにしておたくになり金持ちになったかを探るのがこの本。アメリカによくある立志伝ものだが，相手がビル・ゲイツであるだけになかなかひとすじ縄ではいかないのだ。

前半と後半に分けて読むとわかりやすい。

前半はゲイツの小学生時代からマイクロソフト社を作るまでの大天才時代。ここではゲイツの神童ぶりと同時に，コンピュータ黎明期からパソコン誕生までの歴史がこと細かに述べられていて興味深くかつ資料としても面白い。ゲイツがいかにしてコンピュータに触れ，いかにしてその世界にのめりこんでいったか。そして，世界最初のコンピュータキット「Altair」と，それ用のBASICを作ったビル・ゲイツとその友人の話はほかではなかなか読めないネタであって，古くからのパソコンファンにはたまらないはずだ。そもそも，かなりいい環境で育っているというのはいえる。

後半は，ゲイツ躍進マイクロソフト急成長の大人になった神童の話。MS-DOSの原型を買い取って16ビットパソコンのOSを獲得した話から，Windowsでは大いに苦労したという話などなどが語られるなか，ゲイツへ恨みつらみを持つ者の増加現象もうまくとりあげられている。マイクロソフトがひとつモノを作り出すたびに，ひとり敵が増える，って感じだ。「ゲイツは凄いけどやなヤツ」っていう論調がどんどん強くなっていくのだ。

面白いのは，ゲイツはパソコンおたくである以上にビジネスおたくだということ。そして，マイクロソフトは全ジャンルのソフトウェアでトップをとるつもりでいて，そのためならどんな手段も厭わないということ。現実はそのとおりになり，マイクロソフトは周囲に喧嘩を売りながら手を広げている。なんともまあ恐ろしいことである。ぶるぶる。（K）

ヒューマン
インタフェースの
未来
野々垣旦/小林康人/森田修三編著
富士通経営研修所刊
☎03(3730)3250
四六判　218ページ
1,800円(税込)

いまコンピュータは，もはや単なる「計算機」ではない。望まれているのは，手軽で頻繁に使える「多機能の道具である」ということだ。では，「だれでも，いつでも，どこでも」使われるようになるには何が必要なのだろうか。

本書は３部構成により，おのおのの専門家がそのことを考察する。ハードウェアや使用環境におけるインタフェイス，ソフトウェア機能におけるインタフェイス，そして今後の動向を決定する概念を吟味したうえで，技術や方式の提言を試みる。

コンピュータとの関係を人間主体のものにするためには，まだまだ手探りの必要があるようだ。

劇場としての
コンピュータ
B・ローレル著
遠山峻征訳
トッパン刊
☎03(3295)3461
A5判　263ページ
3,600円(税込)

コンピュータが，単なる機械ではなく，真に人間の道具となり，コミュニケーションの手段として有効になるのは，技術という問題から解放されたあとのことだという。その考えを根底に著者は，人間とコンピュータの関係，さらにはそれを用いた人間どうしのコミュニケーションについて，演劇論を持ち込んで多岐にわたって語っている。

簡単にいえばバーチャルリアリティを解説した本なのであるが，展開されているのは抽象論が多い。決して「読みやすく」はないのだが，ヒューマン・インタラクション(対話)についてのアプローチを，独特の視点で行っている点は重要である。

QUESTION and ANSWER

X-BASICでプログラムを組んでCコンパイラでコンパイルしているのですが，バックグラウンド面とグラフィック面の優先順位のつけ方がわかりません。スプライトだけだと定義しきれないので，どうしてもグラフィック面を使わなければならないのです。あとスプライト面とグラフィック面の優先順位のつけ方もよろしかったら教えてください。よろしくお願いします。

埼玉県　服部　博文

X-BASICで画面間のプライオリティ（表示優先順位）を指定することはできませんが，I/Oポートを直接操作すれば画面間プライオリティの変更は可能です。画面間プライオリティはビデオコントローラのレジスタ1の上位8ビット($E82500)で決められます。各ビットの意味を図1に示します。プライオリティは2ビットで表し，

00>01>10

の関係になります。異なる画面に同じプライオリティや“11”を設定することはできません。標準では“6”(0b00000110)が設定されていますので，

スプライト＋BG＞テキスト＞グラフィックとなっています。

C言語でプライオリティを変更するサンプルプログラムがリスト1です。13行に画面間プライオリティの設定データを記述します。たとえば，グラフィック＞スプライト＋BG＞テキストにするなら13行は，

＊VCR1＝0b011000;

とします。これを参考にしてBCで変換したCのソースに手を加えてみてください。

1992年6月号の付録ディスクに収録されたAPIC_LOAD.Sをアセンブリ言語から使う方法を教えてください。自分でいろいろ試してみましたが，どれも画面モードが切り替わるだけでなにも起こりません。

愛知県　山中　大呉

APIC_LOAD.Sに定義されている_apic_loadをほかのプログラムから使うには，

```
    pea.l    filename(pc)  ＊ファイル名
    move.l   #0,-(sp)      ＊Y座標
    move.l   #0,-(sp)      ＊X座標
    bsr      _apic_load
    lea.l    12(sp),sp     ＊スタック補正
             ⌇
filename:
    dc.b     'A:¥WOOD.PIC',0
```

のようにスタックにファイルネームが格納されているアドレス，ロードするX，Y座標を指定します。PICファイルはフルパスで指定して末尾に‘0’をつけます。_apic_loadを呼び出したあとはスタックを補正します。

特に注意する点はプログラム実行直後はメモリブロックが最大になっていることです。メモリブロックが最大のままだと_apic_loadがバッファの確保に失敗して異常終了します。山中さんの作ったリストは，ファイルネームをポインタで与えていないことと，メモリブロックの変更をしていないことが原因で動かないのでしょう。リスト2を参考にして頑張ってみてください。

メニュー表示を行いカーソルキーでメニューを選択して実行するサブルーチンを作りました。これがHDのシステムから起動すると動作するのですがFDのシステムから起動すると動作しません。私のプログラムとCONFIG.SYS，AUTOEXEC.BATを送りますので，どこが悪いのか教えてください。

埼玉県　横井　健一

おそらくKEY定義ファイルを設定していないのが動作不安定の原因でしょう。X68000は起動時にCONFIG.SYSファイル中に，

KEY＝キー定義ファイル

があるか，起動ドライブのルートディレクトリにKEY.SYSがあるとファンクションキーやカーソルキーの割り当てを設定することができます。X68000のシステムディスクに梱包されているキー定義ファイル(KEY.SYS)では，

↑　CTRL＋A ($01)
←　CTRL＋S ($13)
→　CTRL＋D ($04)
↓　CTRL＋F ($06)

となっています。ところがKEY.SYSを組み込まない標準のキー定義だと，

↑　ESC J　($1B＋$4A)
←　CTRL＋H ($08)

図1

$E82500　ビデオコントローラ　レジスタ1

bit15			8	7			0
未定義	SPRITE	TEXT	GR	GP3	GP2	GP1	GP0

表示優先順位は00>01>10　(11は設定禁止)

リスト1

```
 1:
 2: #include        "stdio.h"
 3: #include        "doslib.h"
 4:
 5:         char *VCR1 = (char *)0xe82500; /* ﾋﾞﾃﾞｵｺﾝﾄﾛｰﾗreg.1 */
 6:
 7: void main()
 8: {
 9:         int ssp;
10:
11:         ssp = SUPER(0);
12:
13:         *VCR1 = 0b000110;  /* 画面間ﾌﾟﾗｲｵﾘﾃｨ変更 */
14:
15:         SUPER(ssp);
16:
17: }
```

リスト2

```
 1: *
 2: * アセンブラからAPIC_LOAD.Sを使う
 3: *
 4:         .include        doscall.mac
 5:         .include        iocscall.mac
 6:
 7:         .xref           _apic_load
 8:
 9:         .text
10:         .even
11:
12:         lea.l   16(a0),a0
13:         suba.l  a0,a1
14:         pea.l   (a1)
15:         pea.l   (a0)
16:         DOS     _SETBLOCK               * メモリブロック変更
17:         addq.l  #8,sp
18:
19:         move.w  #12,d1
20:         IOCS    _CRTMOD
21:         IOCS    _G_CLR_ON
22:         move.l  #0,-(sp)                * ロードY座標
23:         move.l  #0,-(sp)                * ロードX座標
24:         pea.l   filename(pc)            * ファイルネームへのポインタ
25:         bsr     _apic_load
26:         lea.l   12(sp),sp
27:         DOS     _EXIT
28:
29: filename:
30:         dc.b    'i:¥a.pic',0            * PICファイルネーム
31:         .even
32:
33:         .end
```

→ ESC S ($1B+$53)
↓ ESC U ($1B+$55)
となります。INPOUT()関数は最初の1文字のキーコードを返しますので，結果として標準キー定義では←以外，すべて$1Bを返すことになります。

解決策としてはKEY.SYSを起動ドライブのルートディレクトリに置くことでしょう。もっとも多くの人に使ってもらうようなプログラムなら，KEY.SYSの定義内容によって誤動作するようなプログラムは歓迎されません。環境に左右されないプログラムを書きたいなら，INPOUT()の代わりにBITSNS()を使うといいでしょう。なおKEY.SYSはHuman68kシステムディスクのBINディレクトリにあるKEY.Xで新規作成/更新することができます。

MAGIC.FNCのMAGIC_SCREENで，拡張モードにする方法を教えてください。ソースリストを変えればいいのはわかりますが持っていません。　兵庫県　吉井　剛

MAGIC.FNCはMAGIC Ver.1.0で使用することを前提に制作されました。そのためVer.2.0以降で拡張された機能を1991年7月号に掲載したMAGIC.FNCで使うことができません。ちょっと面倒ですが，MAGIC_AUTOコマンドがバッファに格納したデータをMAGICのコマンドと解釈して実行することを利用すれば，Ver.2.0以降で拡張された機能を利用することができます。例として256×256 256色2面を拡張モードで設定する手順を示します。まず1991年9月号71ページに掲載された表を見ると，希望の画面モードを設定するにはコマンド番号$0011にパラメータ0＋$100を与えればいいことがわかります。プログラムにすると，

```
10 magic_flush()
20 magic_init()
30 screen 0,2,1,1  /* G画面初期化
30 dim int scr(3)={&H11,256+0,&HF}
40 magic_putbuf(1,scr)
50 magic_seek(1,0,0)
60 magic_auto(1)
```

となります。

MAGIC_AUTOコマンドはグラフィック画面が初期化されていないと使えません。またMAGIC_AUTOコマンドはバッファに終了コマンド($000F)が現れるまで，エラーチェックを行わずに連続実行します。コマンド列の最後に必ず$000Fを置くことを忘れないでください。

Z-MUSICについて質問がありましたのでペンを取りました。現在U-20をX68000につないでいるのですが，U-20を立ち上げZMUSIC.Xを起動すると，楽器が“MIDI Buffer Full”を起こしてハングアップしてしまいます(U-220でも同じでした)。またMUSICDRV.Xでデータを演奏させようとしてもハングしてしまいます。楽器側の設定に問題があるのでしょうか。

いままで専用シーケンサを使用していたため，DTMに関しては初心者(？)なので，詳しい解説をお願いします。

追伸

私のPROは15MHz化してあるのですが，Z-MUSICだとOPMが正常に演奏されません(15MHzのとき)。OPMDRV/2.Xだと問題ないのですが，やっぱりタイマ関係でダメなんでしょうか。

埼玉県　加賀谷　匠

MIDI Buffer Fullとは大量のエクスクルーシブメッセージやコントロールチェンジコマンドをMIDI機器に送信した場合，MIDI機器の受信バッファが溢れてしまった状態です。ZMUSIC.Xは起動時に表1のコントロールチェンジコマンドを送出してMIDI機器の初期化を行います。

MIDI規格ではMIDIの通信速度で送られた信号はすべて処理されることが前提になっています。はっきりいえば楽器側の責任ですが，ちゃんと処理してくれる音源のほうが珍しい現状ではメーカーに苦情をいってもしかたないでしょう。

応答の遅いMIDI機器のために最新版のZMUSIC.X（1月中旬現在Ver.1.48）にはエクスクルーシブメッセージを送信したときの待ち時間をn/60秒単位で指定できるオプションがあります。ですが起動時にハングアップするのならこのオプションを指定しても意味がありません。なぜなら起動時はエクスクルーシブメッセージを送信していないからです。

対処法としては，ZMUSIC.Xを初期化なしモード（/n）で立ち上げることで，(i)実行時の初期化は回避できます。起動時には楽器を接続せず，あとは手動で表1の初期化を行ってください。

ちなみにVer.1.10やZMUSIC.X最新版では起動時にU-220がハングアップするようなことはないそうです。

それからZ-MUSICは正常なX68000用に最適化されているのでクロックアップしたマシンでは正しく演奏できません。ver.1.0を発表したときにはクロックアップ改造をしたマシン用の（X68000 XVI24MHz用となっていますが，クロックアップしたマシンすべてに有効）ZMUSIC.Xを作成するためのバッチファイルが付属していました。近々発売される改訂版には収録されるでしょう。

FM音源部分はクロックアップした際に動作不安定になる部分の代表例です。基本的にこの程度の症状にも対処できないような人は改造を行うべきではありません。

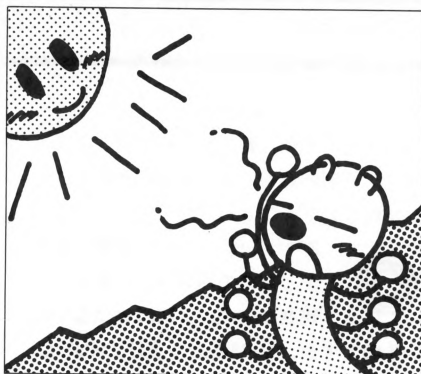
（影山裕昭）

表1

システムリセット($FF)
リセットオートコントローラーズ($Bn,$79,$00)
オムニモードオン($Bn,7D,$00)
モノモードオフ($Bn,$7F,$00)
ローカルオン($Bn,$7A,$7F)
マスターチューン微調整＝中央($Bn,$65,$00,$Bn,$64,$01,$Bn,$06,$40,$Bn,$26,$00)
マスターチューンコース＝中央($Bn,$65,$00,$Bn,$64,$02,$BN,$06,$40)

質問にお答えします

日ごろ疑問に思っていること，どんなことでも結構です。どんどんお便りください。難問，奇問，編集室が総力を挙げてお答えいたします。ただし，お寄せいただいているものの中には，マニュアルを読めばすぐに解答が得られるようなものも多々あります。最低限，マニュアルは熟読しておきましょう。質問はなるべく具体的に機種名，システム構成，必要なら図も入れてこと細かに書いてください。また，返信用切手同封の質問をよく受けますが，原則として，質問には本誌上でお答えすることになっていますのでご了承ください。なお，質問の内容について，直接問い合わせることもありますので電話番号も明記してくださいね。

宛先：〒108　東京都港区高輪2-19-13
NS高輪ビル
ソフトバンク株式会社出版部
Oh!X編集部「Oh!X質問箱」係

FROM READERS TO THE EDITOR

待ちに待った新製品の発売で，X68000の世界もまたひとつ広がりました。新しい風が吹いて，いろんなことが始まる季節。楽しいことや嬉しいこと，つらいことやかなしいこともあるかもしれないけど，それぞれの「冒険の旅」に出発！だね。

◆ついにじゅうくになってしまった。らいねんははたちだろう。　栢　一夫(19)三重県

課題１：この文章を漢字かな交じりで書きなさい(小学５年生以上向)。

課題２：この文を英語で書きなさい。主語は一人称単数とする(中学１年生以上向)。

……なんてね。でも，数のカウントは合ってるから，算数は100点をあげましょう。

◆おや，岡村姉弟はどこへいった？　てっきり連載だと思うほど続いて載ってたもんなぁ。

多田　雅紀(22)愛知県

◆STUDIO Xへお願い。岡村直也さんのために特別のスペースをあげちゃってください。あの４コマ，投稿にしておくのは惜しい。

渡辺　圭(19)北海道

１月号に載っていなかっただけなのに，この反響！　こういう声が多いので，今月は特別サービス豪華２本立てとなりました。みんな満足してくれたかな？

◆１月号の特集「D.I.Y.ハードウェア」はよかったです。ハードウェア工作を本格的に始めようと思えば工具もたくさんいりますが，「作る楽しみ」はなにものにも代え難いです。僕は昔，落としそうになったハンダゴテを思いっきり握ってしまい，掌に大やけどをしたこともありますが(「じゅっ」という音がした)。

天達　雄一(17)京都府

◆D.I.Y.って何ですか？　船越　直弥(20)北海道

おやおや，実はおんなじ質問がいくつか来てしまったので，ここで特別にお答えいたしましょう。D(どーしても)I(いじってみなくちゃ)Y(やだよ～ん)，ということで，“Do It Yourself.”の略，つまり「自分でやれぃ！」ってことですね。

◆昨年４月に就職し，正式に配属されてからの初仕事が，68020ボード用のイーサネットインタフェイスボードの設計でした。いままでZ80しか使ったことがなかったので，いろいろ苦労しました。特に，ローカルメモリ用に作ったDRAMコントローラは大変でした。まだ設計しただけで，動くかどうかわからないのですが……。石上さん，期待してます。がんばってください。

森川　昭夫(24)東京都

１月号の特集は，なかなか大反響でした。特にアクセラレータには皆さん期待大とのことで，石上氏もはりきっています。

◆ちまたでは286マシンが486マシン化しています。VXがRAやDA並みに……。われらがX68000ももうちょっと速くならないかと思っていたところにタイムリーな記事です。失敗したらそれでもいいし，成功したらすばらしい。TeXが速くなるだけでもナイスです。

五十嵐　豊(25)千葉県

◆68020アクセラレータは，ハードウェアよりソフトウェアが大変だと思います。なぜならHuman.sysが動かない，スタックをいじっているようなソフトは動かない，自己書き換えをしているプログラムは動かない，などのことがあるからです。以前，私は68010をX68000に入れましたが，Human.sysが動きませんでした(Human.sysを使わないプログラムは動いた)。どういうふうに解決するのか楽しみです。

久松　愛治(24)東京都

◆アクセラレータの記事はよかったです。何度失敗してもよいですから，ぜひ続けていってもらいたいです。　原田　秀孝(27)神奈川県

◆「アクセラレータを作る」で，「成功する！」といっているのは編集部内で何人くらいいますか？　西池　陽一(15)香川県

えぇと，賭け率は××で参加者は……。なんてことはありません。全員，絶対成功すると信じています(と，いうのも嘘っぽいかな)。まあ，結果はわかりませんが，あたたかく見守っていてくださいな。結果がどうあれ，この連載自身はきっと皆さんの役に立つと思いますよ。

◆つ，ついに始まったアクセラレータ開発計画。私はこの企画が成功するのなら，お百度詣りだろーが，神への祈りだろーが，黒ミサだろーが，丑の刻参りだろーが，何だってやってやるぞ。あ，でもこれが完成する頃には，次期X68000が……(禁句，だったかな？)。

柳井　敏彦(34)愛媛県

ま，それはそれ，これはこれですよね。ところで，石上氏はゾロアスター(拝火)教の信者でいらっしゃいますので，お祈りは儀式にのっとって，火を燃やして「炎の舞」をお願いしますね(嘘ばっかり。信者の人ごめんなさぁい)。

◆なにげなく１月号63ページのSX広辞苑の画面写真を見ていると，おもわずニヤリとしてしまいましたよ。「ぼくの地球を守って」(ぼく地球)ですか……。う～ん，男でこれがわかるのは何人いるのかな～(けっこう多そうな気もするけど)。　黒田　恵一(20)京都府

おお，やっぱり気がつきましたか(知ってりゃ当たり前？)。そう，マンガの登場人物の名前だったのですよね。このことについてのハガキは６通。うち２通が「槐の名前がないのは許さん」とのご意見でした。

◆木蓮，繻子欄，秋海棠……。紀尾井誠氏は「ぼく地球」の読者だったのか。そういえば，10巻以降は読んでいなかった。冬休みが明けたら学校図書館で借りて読むかなぁ。

林　大助(17)神奈川県

最近の高校って，学校図書館にマンガ本も置いてあるのですか？　いいなあ。

◀岡村　直也　兵庫県
仲よく「マチってる」岡村姉弟。元気なふたりの姿にファンも安心，安心。それにしても，でこぼこの姉ちゃんの顔って見てみたいなあ。

◀岡村　直也　兵庫県
さあさあ，豪華絢爛２本立て劇場だよ。姉ちゃんに風邪うつされた弟くん，安心して寝込んでる場合ではないようですね。それがキミの宿命さっ。

◆石田伯仁さんへ。受験勉強がんばってくださいね。後輩より。　　杉山　正仲(16)東京都

これこれ，このページは伝言板ではないぞ。この号が発売される頃は，受験はもう終わってるのかな，それとも，追い込みかな？　石田さんだけじゃなく，皆さん，がんばってくださいね。

◆世間では不況だというのに，私は転職しちゃいました。「転職なんて……」って思ってた私ですが，いざ自分がするといいものです。でも，この転職が最初で最後です(だって，結婚するのに落ち着かなくっちゃ！)。

谷口　博一(26)大阪府

そーか，以前，「日本に帰りたい」っておっしゃってたのは，実は「彼女と離れて寂しい」ってことだったのね。ナットク。ともあれ，しあわせな転職おめでとう。

◆先日，卒論で「To be continued on next sheet」と書くべきところを「To be continued on next stage」と書いた。禁断症状を鎮めるためゲームをした。ふう。嵯峨　進(23)秋田県

そういえば，「continue」という単語って，ゲームで覚えたような気がします。日常での使用頻度が高い(！)から絶対忘れないし。某「でる単」を暗記するより効果的で確実な勉強法(？)かな。

◆いったい，これからの日本の政治はどうなってしまうのだろう。いっそのこと，Oh!X編集部に日本をまかせてしまったほうがよいかもしれない。　　円福　貴光(19)福岡県

そうすると，編集長が総理大臣で，副編集長がナントカ大臣で，スタッフの○○さんは……。そして，Oh!Xは「政府広報」！

◆うちの学校のE先生は学生時代に，日本初のコンピュータをばらして電話交換機を作ったらしい。その後，雑誌でそのコンピュータが行方不明になっているのを知ったとか……。

P.S.　セーラームーンが話題になっていますが，メガドライブのCD-ROMソフト「魔法の少女シルキーリップ」もよいそうです。

程田　勝也(19)茨城県

まさか，「これが変わり果てた姿です」っていって電話交換機を持ってくわけにはいかないでしょうね。まあ，その後も日本のコンピュータはこうして無事に進化してきたわけだし，よかったよかった。

◆ふと鏡を見る。ギャー「荻窪圭」がいる〜。よく見ると(見るまでもないことだが)，寝ぼけまなこの私がいた。髪型と眼鏡を変えようと決意した私であった。そういえば，Oh!Xにはまだ荻窪圭氏の顔写真は載っていないな……。

中内　英裕(28)栃木県

「ギャーとは何だ。けしからん」と，荻窪氏がいうかどうかは不明です(だって，このハガキは氏からは隠しちゃったんだもん)。でも，生活に困ったら彼の「影武者」として生きるって道もあることが判明したじゃありませんか。よかったよかった(？)。

◆1992年12月25日，彼女と別れた……。人生って何ですか？　　黒木　健司(15)大阪府

▼小川　伸輔　宮城県
可愛い彼女はX68000SUPERちゃん。泣いてくれるな、つらいのはボクのほうさ。こんなにクールな顔してても、胸のなかは涙の洪水さ、って？

▼西本　英樹　北海道
LIVE inでおなじみの西本さん、こんなかわいいイラストどーしていままで投稿してくれなかったのお。でも、また送ってくれたら許してあげるね。

1992年12月25日，「彼女と別れ話をする」暇もなく仕事をしていた私と，どっちが不幸でしょう(「彼女をつくる暇もなく」ってのが正しいかな)。まあ，元気出しなよ。

◆ぼくは男です。　　高橋　努(22)神奈川県

よおし，証拠を見せてみい！　……べつに何も見たくないけど。

◆毎年思うのだが，除夜の鐘は108では足りないと思う。あ，それと2月は私の誕生日なのでよろしくどうぞ。　　八木澤　良二(18)栃木県

何がよろしくなのかよくわかんないけど，とにかく，おめでとっ！　ところで，この「ぼんのー」ってどうして108なんてハンパな数なんでしょうね。キリよく128とか256とかだと気持ちいいですよね。すると，邪念の多いアナタは512，私はよいこだから32くらいかなぁ。

◆4月から新しい生活が始まるわけですが，いちばんの問題は，いまよりは確実に狭くなるであろう部屋に，現在ある荷物をどうやって押し込めるか，ということです。たとえるなら，メインメモリ512KにSX-WINDOWを入れようとするようなものではないかと思います。

松前　龍次(19)山口県

う〜ん，部屋にもパソコンにも，どらえもんのポケットとか，ドラゴンボールに出てくるカプセルとか欲しいですよねぇ。

◆1月号の知能機械概論で紹介されていた「ぐりとぐら」。私も小さい頃大好きな絵本でした。思えば人生でいちばん最初に感動(？)した本が，これなのかな？　そのわりには話の内容は完全に忘れてましたけどね。　鹿又　健(23)栃木県

子供の頃の感動って，具体的なことはなんにも覚えてなくってもそのエッセンスが自分のなかに溶け込んでいるような気がしませんか？　潜在的という大げさだけど，けっこう根源的なところで強い影響を受けていたりして。

◆有田隆也さんは，まわりの目を気にせずに絵本を買いあされるようなお年なのでしょうか？すごいですね(何が？)。

小山内　将剛(20)青森県

ということは，キミはまわりの目を気にしながら「何か」を購入しておるのかね？　ある人は，えっち本を買うときに「領収書ください」といえば，売り場のおねえさんの見る目が変わる，っていってましたけど。

◆試験勉強中の徹夜明けに朝からテレビで「ウゴウゴルーガ」を見たら，一日中頭が変でした。

長野　慎太郎(17)東京都

「ウゴウゴルーガ」は評判になっていますね。寝ぼすけの私は徹夜明けにしか見たことがありませんが，頭は大丈夫……って，試験勉強じゃなくてゲームにはまってただけだからかな。

◆期限切れの胃薬は鼻につんとくる。

谷口　浩史(19)北海道

胃の薬でおなかをこわしたりして。

◆15歳の女の子に手を出したら犯罪になるのでしょうか？　　深沢　享廣(20)東京都

どこに出すかにもよる……!?

◆ねむい……。　　堀川　英雄(23)大分県

◆なにかとしんどい。　石田　良一(18)兵庫県

ああああああ。みんなぁ，しっかりしてくれよぉ〜。春はもうすぐじゃないかー。

◆知らなかった。486マシンより10MHzのX68000のほうがウィンドウの動作が速いなんて。そういえば以前，荻窪氏が，MacintoshやIBM互換機と比べても，それほどスピードに違いはないっていってたっけ。もしかしたら，X68000ってそんなに遅いマシンじゃないのかな。私はMacintoshやDOSマシンを使ったことないから，わかんないのだ。　青島　一高(24)静岡県

う〜ん。何をするかにもよりますけど，使ってて感じるスピードって，CPUパワーやクロック数だけの問題じゃありませんしね。そもそも，陸上選手じゃないんだから速けりゃいいってもんじゃないし。大事なのは何がどうできるかだよなぁ。この子はスタイルもいいし，声だって悪くないし，頭もいいし……。うう，なにやら「親ばか」みたいな発言になってきた……。

◆日頃，テープの音が悪いと思っている人はいませんか？　場合によってはヘッドの下のネジを調節するだけで音がよくなります。これはへ

ッドの角度(アジマス比というらしい)を調節するもので，音がクリアになります。テレビなんかも，怖がらずに分解して調節するだけできれいになるはずです。ぜひお試しください。

竹内　大祐(17)長野県

「怖がらずに」ってのは難しいけど(ハードに弱いワタシ)，知ってる人にちょっと聞いてみるのもいいかもしれませんね。簡単な調整でぐんとよくなることも多いのかも……。

◆12月号で「わかりやすい音源のマニュアルを」との碓井さんへ。SC-55のがいちばんいいと思います。それ以外に，というなら音楽の友社「コンピュータ＆MIDI2・テクニカルブック」がおすすめです。GSとGMについては触れていませんが，Z-MUSICでMIDIを扱うにはうってつけの内容です。それから，オーバーテイクをMIDIに切り替えるとき，SC-55のLCDを見てみてください(もう知ってますか？)。いやあ，さすがズームですねぇ。感心しました。

佐藤　仁(24)静岡県

これからMIDIに「入門」しようとしている方，参考になったでしょうか。ちなみにRolandのサービスセンターに問い合わせれば，各機種のマニュアルを部品扱いで購入できるそうです。

◆いっこうにプログラミングの技術が向上しない私を尻目に，メモリ増設，MIDI装備など着々と進歩してきた我がX68000は，最近「プロテクト破り」(？)なる技を会得したようだ。ライトプロテクトをものともせず，ディスクに書き込みしてしまう。さすがはX68000，あなどり難し。

新野　太郎(20)東京都

もしかして，なんか「特殊改造」してるんじゃ……。

◆次期X68000の予約受け付けを始めたショップがあるらしいが，何もわかってないのに予約する人がいるのだろうか。やっぱりいるんだろうな，変なヤツは。　豊田　貴広(22)福井県

パソコンコレクターとか，待ちくたびれて何がなんでも買うぞ，って人とか，お金がありすぎて困っている人とか……。う～ん，どんな人なんでしょうね。

◆通信やってる友達(恩人。PC-9801ユーザー)にTeXを落としてもらうことになりました。ところで，彼がなぜ恩人であるかといいますと，私がサークルの飲み会で突っ走ってリバースモードに入ろうとしたとき，いきなりポケットから黒いゴミ袋を出してくれたうえに，家に泊めてくれたのでした。なんでゴミ袋があったかは謎ですが……。　清水頭　武信(21)東京都

ゴミ袋を常備していると，いろいろ便利なんですよ。突然お花見がしたくなったときに敷物にする，会いたくない人を見かけたときに頭からかぶってゴミのふりをする，寒くなったら穴を開けて着る，よそん家で巨大なおみやげをもらっても平気だし，ほら，リバースモードの人の恩人にもなれるでしょ，ねっ。

◆昨夜，悪夢というのにふさわしい夢を見た。NHKの番組でアメリカのCGを流していて，その内容が，Tシャツ姿の数人の男の背中にスパークが走って，そこから血がドクドク噴き出してきて，肩から腕が落ちて胴がねじ切られて，途中から入ってきた男にいたっては，いきなり頭が割れて血だらけになって天井にきりもみしながら脳天から突っ込んで行く，という夢だった。自分としてはこういうものは大きらいで，いまでも胃がむかついているんだけど，M.N.M.ソフトウェアの「Traum」には期待している。

音羽　進(18)宮城県

最後の文の文脈は「？」。「Traum」は夢は夢でも「悪夢」じゃないから大丈夫ですよね，きっと。楽しみですね。

◆ふだんからZ-MUSICのことを考えている私はある晩，夢を見ました。場所はデパートのおもちゃ売り場らしきところで，客や店員もいず，あたりは真っ暗でした。私はそのなかを歩いていると，顔は見えないけどある男の人がちらっと見えたので，追いかけてみると，向こうも足が速くなって逃げるではありませんか。私は「西川善司さんだな」と，ピーンとひらめき，逃げる彼を追っていくと，西川さんは急に立ち止まり，そして私に向かっていいました。「もう俺にかまわないでくれ」と。何か奇妙な夢でした。P.S.私はZ-MUSICがVer.10000になるまで応援してます。がんばってください。

牧野　裕二(19)埼玉県

まったく，西川氏も人の夢に登場したりしてないで，さっさと原稿書いてほしいものですよね。ぶつぶつ。いや，もしかしたら彼はこうやって読者の人々を訪問してZ-MUSICの宣伝をしてるのかもしれない，と思っておきましょう。

◆今年の初夢はMIDI内蔵のX68000でした。前面にスイッチがいっぱいついていて，マンハッタンシェイプ型とPRO型があり，3.5インチのMOを2ドライブ内蔵していたような……。でもこんな夢を見るなんて，封印しているからだろうな，やっぱり。　松井　雄吾(18)北海道

昔の人は，夢に出てくるというのはその人に想いを寄せられているからだ，と考えていたそうです。かまってもらえない松井さんのX68000が「美人」に化けて，恨み言をいいに出てきたのかもね。

◆テンキーの「3」がこわれて1年以上たつ。いちどは分解して直したものの，最近ではまったく反応しなくなった。ボタン1個の修理代ってどれぐらいだろう。　部家　彰(19)徳島県

「3」って使用頻度が高いんでしょうか？昔，ある会社で，ゲームに使うキーだけがボロボロになったパソコンを見たことがありますが……。当然ながら仕事用パソコンなんだけど，アワレな姿だったなあ。

◆修学旅行で韓国に行ってきました。そのとき，生まれて初めて飛行機に乗りました。離陸する前は，たいしたことないやん，と思っていましたが，滑走路を走りだした瞬間，そんな考えはどこかにいってしまいました。加速しだしたときにかかるあのG，あのエンジン音，もうやみつきになりそうです。スチュワーデスさんはきれいだし，行きも帰りも窓側の席だったので，いうことなしだったんですが，ただひとつ残念だったのが機内食をおっさんが持ってきたことでした。　中山　忠雄(17)滋賀県

飛行機代を値切ったら，そういうとこで差がついた……とか。

◆最近忙しくてパソコンに触れません。グスン。でも半年前，もっとすばらしいものを手に入れることができました(機械とかではなく，よき理解者)。　石塚　潤(21)茨城県

ああっ，また今月も「おのろけ」のハガキが来てしまいました(毎月，必ず何枚かあるんだよなぁ，これが)。皇太子妃も発表されたことだし，高原氏の「予言」どおり，今年は結婚ブームになるのかな？

◆入浴剤と間違えて，ポリデントを入れてしまいました。どうなったかはあえて報告しません。自分で確かめてみるのもまた一興かと。水でもかぶって反省します。ぐしぐし。

岩瀬　貴代美(21)福岡県

おおっ。こんなところに(で)氏のお仲間が(2月号のショートプロを見てね)。でも，まさか入浴はしませんでしたよね？　してたら，(で)氏と同レベル！　反省するなら

◀秋野　潤　静岡県
「ソーサリアン」ですね。繊細な線には，独特の魅力がありますね。トーンもベタもなしのまこういうタッチで描き続けるのもいいかも。

◀横井　賢一　富山県
「ストライダー飛竜」なんですけど，あれ？　あんなところに飛竜ちゃんが……。戦うときでも彼女たちには，つい手加減しちゃったりして。

この季節，水じゃあ寒いから，かぶるのはお湯でいいですよ。……私って女の子には甘いのよね。

◆平日のPM5：30すぎに，NHK教育テレビで，音楽ファンタジー「ゆめ」を放送しています。クラシック音楽が流れ，画面はCGアニメーション。DōGAでCGAを目指す私には，いいテキストです。　鈴木　晴司(28)新潟県

それは，1992年7月号の特集のなかで紹介した「DREAM」のことですね。このCGアニメーションは「響子inCGわ～るど」の寺尾響子さんが担当しています。まだ見たことがない人はぜひ見てくださいね。ちなみに，前ページで話題になった「Traum」のグラフィックも寺尾さんの担当です。

◆猫の前足の裏(まんじゅう？)をくすぐってみる。反応がない。今度は後ろ足に挑戦。ピクピクッ。おー，いやがってるいやがってる。こいつは面白い。後足はくすぐったいとは，まるで人間と同じではないか。みなさんもまわりに寝ている猫がいたら，ぜひお試しあれ。

中島　民哉(22)埼玉県

中島さんちの猫って，ピアノで驚かされたり，足の裏で実験(？)されたり，ちょっと同情しちゃうなあ。まあ，私の友人んちの猫なんか，足の裏(「ニクキュウ」っていうのかな)に落書きされてたけど……。

◆CARDDRV.Xはどうすれば手に入りますか。Z-MUSIC(SC-55)とは何ですか。――初心者なので。　友菊　学(15)千葉県

CARDDRV.Xは，Oh!X1991年1月号の付録ディスクに収録されています。Z-MUSICは音源ドライバで，Oh!X Books「Z-MUSICシステム」として発売されました。現在は品切れで，改訂版を近日中に発売する予定ですが，詳細についてはまだ決まっていません。ドライバ本体のみのバージョンアップ版はOh!X1992年6月号の付録ディスクに収録されています。バックナンバーの購入については，在庫を確認のうえ，お近くの書店にご注文ください。バックナンバーの案内は今月号の64ページにあります。また，SC-55は，Roland製のMIDI音源で，LIVE in'93のページでZ-MUSIC(SC-55)とあるのは，Z-MUSICを使ってSC-55を鳴らす，ということです。

▲佐田　匠　千葉県
こちらも「ストライダー飛竜」ですが，うって変わってかっこいい飛竜くん。この差はきっと作者の「愛情」の違いなんでしょうねぇ。

ぼくらの掲示板

●掲載ご希望の方は，官製ハガキに項目(売る・買う・氏名・年齢・連絡方法……)を明記してお申し込みください。
●ソフトの売買，交換については，いっさい掲載できません。
●取り引きについては当編集部では責任を負いかねます。
●応募者多数の場合，掲載できないこともあります。
●紹介を希望されるサークルは必ず会誌の見本を送ってください。

売ります

★アナログスティック「CZ-8NJ2」を10,000円くらいで売ります。値下げ可。箱，付属品すべてあり。手渡しを希望します。連絡は往復ハガキでお願いします。〒173　東京都板橋区板橋3-22-2-403　池田　健一

★Roland MIDI音源モジュール「MT-32」を送料込み25,000円で売ります。マニュアル，付属品はありますが，箱はありません。連絡は往復ハガキでお願いします。〒510-03　三重県安芸郡河芸町上野1664-1　寺本　篤司(19)

★X68000用漢字ドットプリンタ「CZ-8PK6」を30,000円で売ります。箱なし，インクリボン新品，マニュアル，ケーブルありです。連絡は往復ハガキでお願いします。〒321-43　栃木県真岡市東大島773　安立　宣弘(27)

★X68000 ACE，PRO用1Mバイト増設RAM「CZ-6BE1-A」を10,000円前後，HAL研ファインスキャナ「HGS-68」を20,000円前後，Roland製「はなうたくんCP-40」を17,000円前後で売ります。それぞれ箱，保証書，マニュアル，付属品すべてあり。連絡は往復ハガキでお願いします。〒982　宮城県仙台市太白区大野田字土手前1A-202　西川　勲(36)

★アナログスティック「CZ-8NJ2」を13,000円くらいで売ります。新品同様，付属品，箱すべてあり。連絡は往復ハガキでお願いします。〒299-02　千葉県袖ヶ浦市のぞみ野55-10　今野　道洋(19)

★24ドット漢字プリンタ「CZ-8PK6」を送料別20,000円で売ります。連絡は往復ハガキでお願いします。〒440　愛知県豊橋市新吉町30　竹内里奈

★X68000 XVI用増設メモリ「CZ-6BE2A」を1個，「CZ-6BE2B」を2個，それぞれ送料込み28,000円で売ります。ばら売り可ですが，まとめて買ってくれる方を優先します。連絡は往復ハガキでお願いします。〒272　千葉県市川市国府台4-7-29　水野　一雄

★X68000用MIDIボード「SX-68M」＋Roland MIDI音源モジュール「MT-32」を30,000円で売ります。「MT-32」の箱はありませんが，そのほかの付属品マニュアルはすべてあり。連絡は往復ハガキでお願いします。〒140　東京都品川区東品川3-32-29-2-305　新野　崇仁(19)

買います

★データレコーダ「CZ-8RL1」（ケーブル付き）を20,000円で買います。連絡は往復ハガキでお願いします。〒799-26　愛媛県松山市太山寺町2384-41-17　芳野　聖吾(24)

★X1用カラーイメージボードII「CZ-8BV2」を付属品付き（ディスク版）を13,000円で買います。連絡は往復ハガキでお願いします。〒065　北海道札幌市東区東苗帆9条2丁目12-11　小島英二(21)

★MIDIボード「CZ-6BM1」または同等品を送料込み10,000円で買います。連絡は往復ハガキでお願いします。〒737　広島県呉市弥生町6-33　谷本和生(39)

バックナンバー

★Oh!X1991年1,5月号を送料込み各1,500円で買います。多少の切り抜きは可ですが，付録ディスクのないものは不可です。まずは，官製ハガキで連絡をしてください。〒500　岐阜県岐阜市茜町64-1　伊藤　治(17)

★Oh!X1989年9,12月号，1990年5,8,10,11月号，1991年2月号を送料込み各1,500円で買います。Oh!X LIVE inの記事が無事であれば多少の切り抜き，汚れはかまいません。連絡は往復ハガキでお願いします。〒693　島根県出雲市大津町426-9　伊藤　健一(17)

★Oh!MZ1986年7,11月号，1987年1～3,9月号，Oh!X1988年4,6,10,12月号を送料込み各1,000円で買います。また，エイプ編集「マザー百科」（小学館）を送料込み2,000円で買います。いずれも切り抜き不可。連絡は官製ハガキでお願いします。〒982　宮城県仙台市太白区ひより台22-6　郡山　知行(17)

編集室から from E・D・I・T・O・R

DRIVE ON

このコーナーでは，本誌年間モニタの方々の意見を紹介しています。今月は1月号の内容に関するレポートです。

●68020アクセラレータボードは，以前トランジスタ技術に載っていましたね（ただしPC-9801用）。雑誌の性格上，トランジスタ技術の記事はちょっと難解でした。ソフトを理解できてもハードはあんまり……という人が結構な割合でいるOh!Xでこのようなことをやるなら，とてもとてもわかりやすく載せなくてはならないと思います。その点，1月号の特集では親切な図も掲載されていたし，本文中でも細かく説明してあったのでよかったです。あとは，配布するぐらいの余裕をもってから特集を組んでほしかったですね。

村上　洋樹(17)　X68000 SUPER，PC-G813　埼玉県

●1月号の特集にあった「68020ボードの構想」は疑問に思うところがありました。まず，完結していないということ，素人が手を出せるものではないということ，新型のX68000が発表されてしまえば，忘れ去られてしまうであろうことが予測できること，対応ソフトの問題……いずれを取ってみても興ざめであり，なぜこのようなことを始めたのか理解に苦しんでいます。ただ，総合的な評価は連載が終了してからということになりそうですが，いくら「パーソナルコンピューティング」とはいえ，限度を超えたものと思っています。特集の中では「ラジコン玩具を動かそう」の記事が，いちばん肩が凝らずに楽しめました。パソコンを破壊するということもなさそうですし，遊びの要素もふんだんでこれならやってみようと思いました。外部関数サポートもいいですね。

湯沢　聡(29)　X68000，XIturboIII，MZ-2531/2861，PC-1360K，MSX/MSX2，PC-6601　埼玉県

●特集の「68020ボードの構想」は，現在の処理速度を嘆いているX68000ユーザーにとって，嬉しい企画ですね。いつか速いMPUを載せたX68000が出るかもしれませんが，それでも現在X68000を使っている人たちにとって，一時の夢(?)を見せてくれます。完成されたときには，ぜひソフトバンクからキットを発売してほしいですね。

山田　智広(21)　X68000 SUPER　神奈川県

●新製品紹介の「サンダーワード」には期待しています。しかし，ワープロは書きやすいだけじゃだめなんです。いくら「一太郎」がバカでのろくてくだらない制御体系でも，あれにはそれなりの表現力と多様なプリンタに対応している，という利点があります。私としては，ver.2.0ぐらいでアウトラインフォントと表現力を大幅に備えてもらえると，再びワープロ作業をX68000に戻せるでしょう。

内藤　陽一(26)　X68000，PC-9801NS/E　東京都

●「X68000マシン語プログラミング」についてですが，1月号の「Human68k ver.2.0の機能」はたいへん興味深く，読みごたえのある内容でした。実際のところ，Human68k ver.2.0はver.1.0の頃に比べて，かなりの面で機能拡張が行われているのだと改めて理解でき，とても嬉しく思いました。逆にHuman68k ver.2.0の弱点というか，不備についても詳細に述べられているのもよかったです。そして，ハードディスクなどの入出力関係ではとても参考になりました。次回の後編を大いに期待させてくれます。

藤田　康一(22)　X68000 PRO　静岡県

●1月号の「X68k Programming Series」には驚きました。GCC用のライブラリまであるとは……X68000は本当にユーザーが作り上げてきたマシンなんだな，と実感しました。しかし，3年前，あり金はたいてXCver.2.0を買った私の立場はどうなるのでしょう。駅から自転車でパッケージを運ぶのも大変でした。今回の記事を見て，ふとそのときのことを思い出してしまいましたが，考えてみれば喜ばしい悔しさといえます。また，私は以前GCCを手に入れたことがあるのですが，何がなんだかまったくわかりませんでした。使い方や環境など，ドキュメントを見ても不明な点が多く，バージョンが入り乱れたライブラリを前に何もすることができないでいたのです。そういった意味では「ドキュメントの整備」というのはとても助かります。これであきらめていたGCCの世界へ，再び入っていけるでしょう。

宍戸　輝光(19)　X68000 PRO，MSX2　東京都

ごめんなさいのコーナー

1月号　Oh!X LIVE in '93

P.68　リスト1の「ムーンライト伝説」のカウンタ表示が掲載されていませんでした。カウンタ表示は，今月号Oh!X LIVE in'93の115ページに掲載されていますので，参考にしてください。先月号に引き続き，ご迷惑をおかけしましたことをお詫びいたします。

2月号　よいこのSX-WINDOW講座

P.46　リスト2で正しくない記述がありました。リスト2の129～131行をリスト1に書き換えてください。

2月号　(で)のショートプロぱーてぃ

P.118　リスト1の差し換え部分であるリスト2の行番号が間違っていました。正しくは，リスト1の900行目からリスト2を打ち込んでください。

リスト1

```
    if ( myRes != cuRes ) {
      RMCurResSet(curRes);    /* カレントリソースを元に戻す */
    }
    return 0;
}
```

バグに関するお問い合わせは
☎03(5488)1311(直通)
月～金曜日16：00～18：00

お問い合わせは原則として，本誌のバグ情報のみに限らせていただきます。入力法，操作法などはマニュアルをよくお読みください。
また，よくアドベンチャーゲームの解答を求めるお電話をいただきますが，本誌ではいっさいお答えできません。ご了承ください。

X68030って おいしいかな しょっぱいかな

▼X-BASICを学ぶうえで避けて通れないもの，それが関数です。構造化という性格をもつX-BASICの世界では基本といえる関数。この概念を理解できないでいると，必然的にX-BASICを理解できないことになってしまいます。結局，どんなプログラムでも関数の集合で成り立っているのですからね。そこで，今月の特集ではひとつの役割をもった関数の作成，拡張，応用方法を探ってみました。X-BASICは実行速度に不満があるとはいえ，インタラクティブという環境を生かせば，まだまだ可能性が広がるでしょう。

▼ついに登場した32ビットマシンX68030。夢を超えた，という衝撃的なデビューから6年たった現在，今度はメインMPUを68EC030に代え，システムの大幅な変更によって再び生まれ変わろうとしています。

今回は，まだまだ情報が少なく，製品スペックのみの紹介という感じでした。互換性やシステム内部など，知りたいことが山ほどあるでしょう。

そんな人のためにも，来月号でもどこがどう変わっているのか，より詳しくレポートしたいと思っています。

▼そして，来月号では毎年恒例1992年度GAME OF THE YEARの発表が行われます。ぼちぼちとアンケートハガキの集計結果が出ようとしています。どの作品がどんな賞を受賞するか，来月号をお楽しみに。

▼さて，今月号にはアンケート用紙が同封されています。これは，5月号で行われる「言わせてくれなくちゃだワ」で使われるものです。言いたい放題，好き勝手なことをいえる読者参加の企画として，もう8回目を迎えます。例年以上の盛り上がりを見せるためにも，読者の皆さんの協力が不可欠です。ガンガンアンケートを返送してくださいね。

▼「大人のためのX68000」「吾輩はX68000である」「よいこのSX-WINDOW」は著者多忙のためお休みさせていただきました。楽しみにしていた方，本当にごめんなさい。

投稿応募要領

- 原稿には，住所・氏名・年齢・職業・連絡先電話番号・機種・使用言語・必要な周辺機器・マイコン歴を明記してください。
- プログラムを投稿される方は，詳しい内容の説明，利用法，できればフローチャート，変数表，メモリマップ（マシン語の場合）に，参考文献を明記し，プログラムをセーブしたテープ（ディスケット）を添えてお送りください。また，掲載にあたっては，編集上の都合により加筆修正させていただくことがありますのでご了承ください。
- ハードの製作などを投稿される方は，詳しい内容の説明のほかに回路図，部品表，できれば実体配線図も添えてください。編集室で検討のうえ，製作したハードが必要な場合はご連絡いたします。
- 投稿者のモラルとして，他誌との二重投稿，他機種用プログラムを単に移植したものは固くお断りいたします。

あて先

〒108 東京都港区高輪2-19-13 NS高輪ビル
ソフトバンク出版部
Oh!X「㋢㋮㋯」係

SHIFT・BREAK

▶馴れとは恐ろしいもので，240Mバイトのハードディスクを狭いと嘆き，9600bpsのモデムを遅いと感じるようになってしまった。だが，こういった愚痴は嫌味にしか聞こえないらしく，誰も同情してくれない。どこかにこんな悩みを聞いてくれて，哀れに思ってお金を恵んでくれる親切なお方はいませんかね。世間の風は冷たいなぁ，うんうん。（八）

▶最近Macで描いた漫画をよく見かけます。やはり充実したソフトと整った出力環境が魅力なのでしょうが，いかんせんMacintoshのソフトは高い！　と思っていたら実はX68000＋MATIER＋C-TRACEを使った漫画もあったのです。学研NORAで連載中の「吸血鬼に違いない」。興味のある方はご一読を。私は思わずやる気になってしまいました。（哲）

▶カプコンのストリートファイターIIの基板をターボにしたいとメーカーに問い合わせてみたところ個人ユーザーは対象外とのこと。驚＆怒。でも，結局渋谷のロータス某でそういうサービスをしていることがわかりひと安心。これでわが家もターボだ。しかし，ターボってもう「初心者お断り」的ニュアンスが強いけどいいんだろうか。（善）

▶昨秋に伊豆半島の北川温泉にある海辺の露店風呂に入りました。打ち寄せる波と満天の星を見ながらつかる温泉は最高。ちょうど混浴の時間帯で若い女性もチラホラ。脱衣場ではすぐ隣に女性がいて不必要に緊張してしまい，私は純情なんだと気づきました。視力が悪いと温泉地での楽しみも半減。本気で視力回復センターにでも通おうかな。（H.K.）

▶またAMIGAの話。今年も出たクリスマス・レミングス。難しいけど素敵。でもプレイしたのは年明け……間抜け。レミングス2についた画期的な機能「早送り」。ここはもうひと声「巻き戻し」もほしい。あと一歩で無念のミスが多いからね。ところでAMIGAのストIIはなんとボタンが1個だそうだ。アーケードの移植ではX68000が数段上だと思う。（A.T.）

▶どっかの誰が結婚するとか婚約解消だとかそんなことはどーでもいいのだが，マスコミがマスコミの報道ぶりを報道して，その馬鹿ばかしさをさらしたのには笑。それを見ながら報道の過熱ぶりをコメントするキャスターも笑。それより，アンドレ・ザ・ジャイアントと安部公房が死んだことのほうがショックなのだ。誰も死ぬときは死ぬんだなあ。（K）

▶ゴジラVSモスラを観た。この映画でゴジラの存在意義はどこにあるのだろう。モスラが出てきてバトラと戦うだけでも話は成り立つ。無理にゴジラを登場させるようでは，新ゴジラのシリーズもそろそろ息切れ気味か。それでも子供の受けは非常にいいようなのでこれで正解かもね。私は最後に宇宙空間をはばたいて飛んでいくモスラが興覚めだった。(KO)

▶子供時代を北国で過ごしたせいか，移り住んで何年もたつのにやはりまだ，東京の冬はもの足りない。降る雪や積もった雪のさまざまな表情を目にすることもないし，なによりもあの，冷たく澄んだ空気を呼吸することがない。なんだかはぐらかされたような感じで季節が過ぎてゆく。あんなに，寒い冬は嫌だと思っていたのに，ささやかに身勝手。（ふ）

▶よりによってまた月末にカゼをひいた。先月に引き続き，となると健康管理もきちんとできない自分に，いい加減腹が立つ。思えば年始から頭の中はぐるぐる回っているし，何かと当たり散らすなどろくなことがない。しかも，こうなった原因が明確でないのも不安だ。なんとなく日々が過ぎていく，そんな無気力な現状をとっとと打破したい。（J）

▶最近は真面目に画像を生成したりしていたのだが，ひさびさにゲームをしたらマウスポートの周辺回路が死んでしまった。AMIGA2000の代わりにSE/30を使うしかないが，やはりメインマシンがないと何もできない。お願いだから，早く復活してくれい（といいつつ，面倒臭いのでまだ修理には出していないのであった)。（A）

▶当然予想されたことだがスタッフの反応はさまざま。ま，技術者の前で「もの足りませんね」といった私も私だが。いずれにせよ触っているとほしくなるマシンであることは確かだ。1.6倍のXVIであれだけ違うもんなあ。ところでX68030ではZ-MUSICもMAGICも動かない。25MHzでRAMが4M……ちびっと待ってなさい。(100万円くらいの機種がほしいU）

▶ベールを脱いだX68030。今月はギリギリのタイミングで速報をお届けできた。さて，車などでは高性能車の象徴とされる赤バッジだが，コスト的に赤字だからとの説もある。CPUばかり速い安普請なマシンがもてはやされる昨今，X68030も価格設定には苦慮したようだ。どかーんと売れて黒字になったバッジの色を黒く塗るなんていわないでね。（T）

microOdyssey

昨年から今年にかけ，DOS/Vマシンメーカーが価格競争にしのぎを削り，PC-9801シリーズやその互換機メーカーも思い切った低価格の新製品を繰り出してきた。Macintoshも新製品の連発で，急激に価格を降下させていく。そして，それぞれの機械はマルチメディアということばをキーワードに，何でもかんでもできるようなイメージを抱かせている。パソコン業界はまさに混沌とした状況である。

現在，主役の座を奪い合っているこれらのマシンも，ひと昔前まではシェアの高さを誇りつつ，一定数の欠点が見え隠れしていた。しかし，いまでは目を見張るほど高速なCPUが搭載され，ほとんどの欠点を覆い隠す。

実際，アプリケーションにしてもゲームにしても，なかなかデキるソフトが続々と登場しているようだ。

そういうスゴそうな機械（きちんと使った経験がないので，こういう形容に止めておく）が手に届きそうな価格で売られているのは，決して悪いことではない。しかし，それが混沌を生んでいることも紛れもない事実である。

X68000シリーズの最新機種“X68030”は，そんななかでの誕生をしいられた。スペックは本文ページでご覧いただいただろうが，CPUのスピードと価格ばかりが注目される現状では厳しい評価を下されかねない内容ではある。周りの動きが気になるのはしかたないことだし，マシン自体もある程度，中途半端であることは否めないだろう。

しかし，他機種と比べるのはあまり意味のないことだというのは心に留めておきたい。これと比べてこうだとかいう相対的評価ではなく，的確にどこがどう悪いという絶対的評価を下すべきだと思うのである。そうでないと，どれもが似たりよったりで，可もなく不可もないつまらないパソコンばかりになりかねない。そして，すでにそうした状況は生まれつつある。

同様に価格もあまり重要ではない，と個人的には思っている。自分の気に入ったマシンがあるのなら，安かろうが高かろうが無理をしてでも買うべきで，価格はあとから考慮に入れればいい。本体の魅力が最優先事項なのだ。

こうした点を踏まえながら，スペック表をもう一度見直していただきたい。評価は上がっただろうか。もちろん，下がってしまったという人もいるかもしれない。しかし，それは冷静な評価であり，声を大にして訴えかけてもいい性質のものである。

そういう声はまた新しいマシンへと受け継がれていく。しかし，メーカーは意見をそのまま取り入れるのではなく，その意見に触発されて新しい試みを導入するという経緯をたどるべきであることも付け加えておこう。

世間はすさまじい勢いで変化し，どんどん新しいモノを生み出していく。新しいモノは古いモノや同時に生まれた新しいモノを押し流しながら，すでに追われる立場にある。そんななかでは，何が生き残り，何が消え去るのかを捉えることは難しい。しかも，生き残ったものがいいものであるともかぎらない。こういう状況下では，自分なりの考えをしっかりともっていないと，時代に押し潰されてしまいかねないのではないだろうか。　（A）

1993年4月号3月18日(木)発売

特集1　X68030解体新書

・SX-WINDOW,Human68k進化論

・MPU68030の概要

特集2　ゲーム学校1年生

1992年度 GAME OF THE YEAR

第5回アマチュアCGAコンテスト結果発表

バックナンバー常備店

東京	神保町	三省堂神田本店5F	03(3233)3312
	〃	書泉ブックマートB1	03(3294)0011
	〃	書泉グランデ5F	03(3295)0011
	秋葉原	T-ZONE 7Fブックゾーン	03(3257)2660
	八重洲	八重洲ブックセンター3F	03(3281)1811
	新宿	紀伊国屋書店本店	03(3354)0131
	高田馬場	未来堂書店	03(3209)0656
	渋谷	大盛堂書店	03(3463)0511
	池袋	旭屋書店池袋店	03(3986)0311
	八王子	くまざわ書店八王子本店	0426(25)1201
神奈川	横浜	有隣堂横浜駅西口店	045(311)6265
	〃	有隣堂ルミネ店	045(453)0811
	藤沢	有隣堂藤沢店	0466(26)1411
神奈川	厚木	有隣堂厚木店	0462(23)4111
	平塚	文教堂四の宮店	0463(54)2880
千葉	柏	新星堂カルチェ5	0471(64)8551
	船橋	リブロ船橋店	0474(25)0111
	〃	芳林堂書店津田沼店	0474(78)3737
	千葉	多田屋千葉セントラルプラザ店	0472(24)1333
埼玉	川越	黒田書店	0492(25)3138
	川口	岩渕書店	0482(52)2190
茨城	水戸	川又書店駅前店	0292(31)0102
大阪	北区	旭屋書店本店	06(313)1191
	都島区	駸々堂京橋店	06(353)2413
京都	中京区	オーム社書店	075(221)0280
愛知	名古屋	三省堂名古屋店	052(562)0077
	〃	パソコンΣ上前津店	052(251)8334
	刈谷	三洋堂書店刈谷店	0566(24)1134
長野	飯田	平安堂飯田店	0265(24)4545
北海道	室蘭	室蘭工業大学生協	0143(44)6060

定期購読のお知らせ

Oh!Xの定期購読をご希望の方は綴じ込みの振替用紙の「申込書」欄にある『新規』『継続』のいずれかに○をつけ，必要事項を明記のうえ，郵便局で購読料をお振り込みください。その際渡される半券は領収書になっていますので，大切に保管してください。なお，すでに定期購読をご利用の方には期限終了の少し前にご通知いたします。継続希望の方は，上記と同じ要領でお申し込みください。

海外送付ご希望の方へ

本誌の海外発送代理店，日本IPS（株）にお申し込みください。なお，購読料金は郵送方法，地域によって異なりますので，下記宛必ずお問い合わせください。

日本IPS株式会社
〒101 東京都千代田区飯田橋3-11-6
☎03(3238)0700

3月号

■1993年3月1日発行　定価600円（本体583円）
■発行人　孫　正義
■編集人　橋本五郎
■発売元　ソフトバンク株式会社
■出版事業部　〒108 東京都港区高輪2-19-13　NS高輪ビル
Oh!X編集部　☎03(5488)1309
出版営業部　☎03(5488)1360 FAX 03(5488)1364
広告営業部　☎03(5488)1365
■印　刷　凸版印刷株式会社

1月20日
第1回サポートサービス(無償)開始

日本語ワードプロセッサ

サンダーワード

ThunderWord ver 1.0

あなたはもう**雷語**(サンダーワード)の使い方を知っている!
かな漢字変換は標準FEPの**ASK68K**に準拠
ED.Xと**MicroEMACS**のコマンド体系
X68000ビットマップディスプレイ機能を活用
ルビ・アンダーライン機能
最大32ファイルを同時編集
最大15までの水平分割ウインドウ
フレンドリーな辞書登録機能
プリンタはCZ,ESC/P,NM,PC-PRに対応
縦・横印刷機能、印刷プレビュー機能

商品・通販のお問い合わせは
〒171 東京都豊島区長崎1-28-23 Muse西池袋2F TEL(03)3554-9282 FAX(03)3554-3856

㈱満開製作所

満開の電子ちゃん

作・え 岡村祭

講読方法：定期購読もしくはソフトベンダーTAKERUでお買い求めいただけます。
★定期購読の場合＝購読料６ヶ月分6,000円（送料サービス、消費税込）を、
現金書留または郵便振替で下記の宛先へお送り下さい。
現金書留の場合：〒171 東京都豊島区長崎1-28-23 Muse西池袋2F ㈱満開製作所
郵便振替の場合：東京 5－362847 ㈱満開製作所
●ご注文の際は、郵便番号・住所・氏名・電話番号を忘れずに記入して下さい。
●3.5インチディスク版をご希望の方は、「3.5インチ版」とご指定下さい。
●新規購読の方は「新規」と明記して下さい。なお、特に購読開始号のご指定がない場合は既刊の最新号からお送りいたします。
●製品の性格上返品には応じられませんが、お申し出があれば定期購読を解約し残金をお返しします。
★TAKERUでお求めの場合＝１部につき1,200円（消費税込）です。
●定期購読版と内容が一部異なる場合があります。御了承下さい。
●お問い合わせ先 TEL(03)3554－9282（月～金 午前11時～午後６時）
（なお、定期購読版のバックナンバーについては定期購読の方のみご注文を承ります）

竹下雄一郎（熊本県）

京都への修学旅行中、凶のおみくじを二回も引いてしまった私はすっかりふさぎ込んでいました。そこへ「信じる者は救われる」と友人が勧めてくれたのが、何を隠そうこの「電脳倶楽部」でした。「今行った清水の舞台から飛び降りるつもりで」と、自由時間にタケル設置店に駆け込み、店頭のマシンで試しに起動すると、予想以上の充実度！「変會長の小屋」を読む頃には、私の心はすっかり希望に満ちあふれていたのです。私は電倶教に入門、自動引落とし近日導入と噂される定期購読を、壺のかわりに申込むつもりです。

予約受付中!

32Bit MPU 68030 High Speed 25MHz

New Model X68030

在庫処分・特価放出

X68000 XVI

¥184,000-

X68000 Compact

¥149,000-

X68000 SUPER

¥99,800-

SX-WINDOW上で広辞苑を使う

SX-広辞苑はSX-WINDOW上で動作するCD-ROM広辞苑検索ソフトです。市販されているCD-ROM広辞苑第三版を検索でき、SX-WINDOWの特徴である、マウスオペレーション、マルチタスク、データの引用機能などが利用できます。エディタX等、他のSXアプリケーションとの同時使用もできます。又、複数のSX-広辞苑を同時起動することでの複数項目の同時検索参照や、CD-ROM広辞苑内に納められている色見本、音声、図版等の検索も可能です。

SX-広辞苑(ソフトのみ)	¥19,800-
SX-広辞苑CD-ROM広辞苑セット	¥45,000-

※広辞苑は岩波書店の登録商標です。
※CD-ROM広辞苑(第三版)は岩波書店から発売されている12cmCD版が対象です。
SONY電子ブック用のCD-ROMは御利用になれませんので御注意ください。

好評発売中! X68000 CD-ROM第一弾

Free Software Selection

価格¥5,000-

中身は買ってからのお楽しみ、CD-ROMならではの大容量での内容です。

X68000用　CD-ROMドライブ

KGU-XCD

X68000のSCSIインターフェースに接続するドライブです。
弊社製FreeSoftwareSelectionやSX-広辞苑などが利用でき、
他機種向けのISO9660フォーマットのディスクも参照できます。
ISO9660 Driver/Macintosh™ファイルビューア/CD Play sampleが附属

低金利クレジット　通信販売送料　全国一律￥1,000　長期クレジット可能
※表示価格に消費税は含まれておりません

株式会社　計測技研　マイコンショップBASIC HOUSE
本社／ショールーム／通販部
〒321 栃木県宇都宮市竹林町503-1
TEL 0286-22-9811　FAX 0286-25-3970

《業界No.1の"P&Aメンテナンスサポート》

最高の保証システム

①業界最長の新品パソコン5年保証
（※モニター・プリンター3年間保証!! ※一部商品は除きます。）
②中古パソコンの1年間保証
（モニター・プリンター6ヶ月間保証）
③初期不良交換期間3ヶ月
（※新品商品に限らせていただきます。）
④永久買取保証
⑤配達の指定OK!!（土曜・日曜・祭日もOK!!）
⑥夜間配送もOK!!
（※PM6:00～PM8:00の間 ※一部地域は除きます。）

便利でお得な支払いシステム

①翌月一括払い手数料無料（ご利用下さい。）
②業界No.1の低金利
③月々の支払いは￥1,000より
④9ヶ月先からのスキップ払いOK!!
⑤84回までの分割、ボーナス併用OK!!
⑥カレッジクレジット
⑦ステップアップクレジット
⑧ボーナスだけで10回払いOK!!
⑨現金一括払いOK!!
（※商品・金額ご確認の上、銀行振込・現金書留にてご入金下さい。）

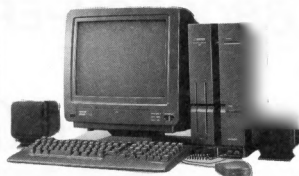
●法人向けリースシステム
業務に最適なシステムを構築します。
損金処理が可能なリース契約をどうぞ。

でっかく 新装!!
（SHARP＝X68030エキスパートショップ）
P&A 新本店OPEN!!

P&A＝X68030 ダブルNEWフェア
《2月18日～3月17日》

注目!!夏のボーナス一括払い手数料（金利）無料（平成5年3月末／4月末／5月末／6月末／7月末のいずれかをご指定下さい。）

フェア記念 第2弾
68030いよいよ登場。購入ダブルチャンス!!

注目!!

■CZ-604D
（ブラック）
定価￥94,800
●14" 0.31mm
●スピーカ、チルトスタンド付

⊙TVチューナー付のモニター（CZ-613Dグレー）に変更の方は￥27,000加算して下さい。
■CZ-613D（グレー）
定価￥135,000
●15" 0.31mm
●TVチューナー、スピーカー、チルトスタンド付

X68030発売記念

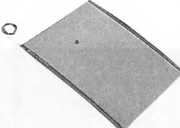
X68030をモニターとセットで／単品で 購入の方

さらに現在お持ちのパソコンと下取り交換されたお客様に期間中もれなく、

①サイバーステック（CZ-8NJ2 ￥23,800）
②CRTフィルター（BF-68PRO ￥19,800）
③X-68000フロッピーアタッシュケース（￥8,000）とクリスタルポルシェ（￥8,000）

以上のいずれかプレゼント!!

①

②

③

●本広告の掲載の商品の価格については、消費税は含まれておりません。

《業界No.1の"P&Aメンテナンスサポート》
最高の保証システム

①業界最長の新品パソコン5年保証
（※モニター・プリンター3年間保証.// ※一部商品は除きます。）
②中古パソコンの1年間保証
（モニター・プリンター6ヶ月間保証）
③初期不良交換期間3ヶ月
（※新品商品に限らせていただきます）
④永久買取保証
⑤配達の指定OK//（土曜・日曜・祭日もOK.//）
⑥夜間配送もOK//
（※PM6：00〜PM8：00の間 ※一部地域は除きます。

便利でお得な支払いシステム

①翌月一括払い手数料無料（ご利用下さい。）
②業界No.1の低金利
③月々の支払いは￥1,000より
④9ヶ月先からのスキップ払いOK//
⑤84回までの分割、ボーナス併用OK//
⑥カレッジクレジット
⑦ステップアップクレジット
⑧ボーナスだけで10回払いOK//
⑨現金一括払いOK//
（※商品・金額ご確認の上、銀行振込・現金書留にてご入金下さい。）

モデム（送料￥1,000 消費税別）

■FMMD-311G
（富士通）定価￥35,800
▶特価￥24,800
（送料・消費税込み￥26,574

■PV-M24V5
（AIWA）定価￥36,800
▶特価￥25,700
（送料・消費税込み￥27,501

■MD-24FB5V
（オムロン）定価￥39,800
▶特価￥23,500
（送料・消費税込み￥25,235

注目!!夏のボーナス一括払い手数料(金利)無料（平成5年3月末/4月末/5月末/6月末/7月末のいずれかをご指定下さい。）

●お近くの方は、お立寄下さい。専門係員が説明いたします。
●本体単品でも受付します。詳しくは、お電話にてお問合せ下さい。
●ビジネスソフト定価の15%引きOK.// TEL下さい。

《増設メモリー&数値演算プロセッサ》計測技研　（送料￥500・消費税別）

①PRKII-02（2M）……定価￥55,000▶特価￥39,800
②PRKII-04（4M）……定価￥90,000▶特価￥67,000
③PRKII-06（6M）……定価￥125,000▶特価￥92,500
④PRKII-08（8M）……定価￥160,000▶特価￥119,000
⑤PRKII-12（2M）……定価￥85,000▶特価￥63,000
⑥PRKII-14（4M）……定価￥120,000▶特価￥89,500
⑦PRKII-16（6M）……定価￥155,000▶特価￥114,500
⑧PRKII-18（8M）……定価￥190,000▶特価￥141,000
⑨MC-68881RC……定価￥38,000▶特価￥27,000

周辺機器コーナー　（送料￥1,000・消費税別）

1 CZ-8NSI……定価￥188,000▶特価￥133,000
2 CZ-6VTI……定価￥69,800▶特価￥49,500
3 CZ-6TU……定価￥33,100▶特価￥23,900
4 BF-68PRO……定価￥19,800▶特価￥14,400
5 CZ-8NM3……定価￥9,800▶特価￥7,200
6 CZ-8NT1……定価￥13,800▶特価￥10,000
7 CZ-6BE2A……定価￥59,800▶特価￥42,800
8 CZ-6BE2B……定価￥54,800▶特価￥39,300
9 CZ-6BE2D……定価￥54,800▶特価￥39,300
10 CZ-6BF1……定価￥49,800▶特価￥35,800
11 CZ-6BP1……定価￥79,800▶特価￥57,000
12 CZ-6BM1……定価￥26,800▶特価￥19,300
13 AN-S100……定価￥36,600▶特価￥26,300
14 CZ-6SDI……定価￥44,800▶特価￥32,500
15 CZ-6BN1……定価￥29,800▶特価￥21,500
16 CZ-6BV1……定価￥21,000▶特価￥15,200
17 CZ-6BC1……定価￥79,800▶特価￥57,000
18 CZ-6BG1……定価￥59,800▶特価￥43,000
20 CZ-6BU1……定価￥39,800▶特価￥28,500
21 CZ-6PV1……定価￥198,000▶特価￥142,000
22 CZ-6BS1……定価￥29,800▶特価￥21,500
23 CZ-8NJ2……定価￥23,800▶特価￥17,500
24 CZ-6BL2……定価￥298,000▶特価￥214,000
25 JX-100S……定価￥89,800▶特価￥44,000
26 JX-220X……定価￥168,000▶特価￥121,000
27 IO-735XB……定価￥248,000▶特価￥152,000
28 LC-10CIH……定価￥598,000▶特価￥459,000
29 CZ-6CSI（674C用）……定価￥12,000▶特価￥8,900
30 CZ-6CR1（RGBケーブル）……定価￥4,500▶特価￥3,600
31 CZ-6CT1（テレビコントロール）……定価￥5,500▶特価￥4,400
32 CZ-6BP2……定価￥45,800▶特価￥33,300

■SX-68MII（MIDI）（サコム）
定価￥19,800（送料・消費税込み￥14,935）
特価￥13,500

■CZ-68HA
●674C用内蔵HD80M
特価￥91,000

FDD（5インチ×2基）
■CZ-6FD5
（シャープ）
（定価￥99,800）
P&A超特価
￥49,800

プリンター（ケーブル用紙付 送料￥1,000・消費税別）

■CZ-8PC5-BK
定価￥96,800
▶特価￥68,500
■CZ-8PK10
定価￥97,800
▶特価￥71,000

カラーイメージジェット

■IO-735X-B
定価￥248,000
特価￥152,000
（送料・消費税込み￥157,590）

X68000専用ハードディスク（外付）（送料￥1,000・消費税別）

■ロジテック
⊙LHD-FM100E
●100M ●19ms
定価￥99,800
▶超特価TEL下さい。

■ロジテック
⊙LHD-FM200E
●200M ●17ms
定価￥138,000
▶超特価TEL下さい。

■システムサコム
⊙HD-J130
●130M ●20ms、富士通、純正ドライブ使用
定価￥148,000
▶特価￥59,500

■ジェフ
⊙GF-240
●240M ●16ms
定価￥148,000
▶特価￥95,000

X68000メモリボード

①SH-6BE1-1M（600C専用）（I/Oデータ）……定価￥25,000
特価￥17,900（送料・消費税込み￥18,952）
②1MB増設RAMボード（ACE/PRO/PROII用）…定価￥25,000
特価￥15,900（送料・消費税込み￥16,892）
③2MB増設RAMボード（拡張スロット用）……定価￥50,000
特価￥31,700（送料・消費税込み￥33,166）
④4MB増設RAMボード（拡張スロット用）……定価￥88,000
特価￥55,200（送料・消費税込み￥57,371）

P&A特選パソコンラック　（送料無料）

①3段￥8,900（消費税込み￥9,167）
●1230(H)×600(D)×650(W)

②4段￥9,900（消費税込み￥10,197）
●1250(H)×700(D)×640(W)

③5段￥12,500（消費税込み￥12,875）
●1310(H)×700(D)×640(W)

全機種＝移動自由（キャスター付）●コードクランプ付（4段/5段）
※5段のみ＝電源コード付（2.5m）（2P）キーボード収納可能

OAチェアー

￥11,500（消費税込み￥11,845）

￥19,800（消費税込み￥20,394）

X68000用ソフトコーナー　（送料￥700・消費税別）

◆Z's STAFF PRO68K Ver.3.0（ツアイト）……定価￥58,000▶特価￥37,500
◆Z's TRIPHONY デジタルクラフト（ツアイト）……定価￥39,800▶特価￥27,000
◆テラッツォ（ハミングバード）……定価￥19,400▶特価￥13,600
◆マジックパレット（ミュージカルプラン）……定価￥19,800▶特価￥14,200
◆たーみのる2（SPS）……定価￥17,800▶特価￥13,000
◆Mu-1 Super……定価￥39,800▶特価￥28,500
◆CMA68K（シティソフト）……定価￥29,800▶特価￥21,800
◆サイクロン EXPRESSα68……定価￥98,000▶特価￥69,000
◆C-TRACE68 Ver.3.0（キャスト）……定価￥98,000▶特価￥68,500
◆C&Professional Pack V3.2（マイクロウェアジャパン）……定価￥80,000▶特価￥57,800
◆ウエットペイント1〜3（ウエーブトレイン）〔各〕……定価￥15,000▶特価￥11,500
◆マチエール（サンワード）……定価￥39,800▶特価￥28,800
◆Windex PRO68（JEL）……定価￥28,000▶特価￥20,500
◆CZ-213MSD MUSIC PRO68K……定価￥18,800▶特価￥13,200
◆CZ-214MSD SOUND PRO68K……定価￥15,800▶特価￥11,300
◆CZ-215MSD Sampling PRO68K……定価￥17,800▶特価￥12,500
◆CZ-220BSD DATA PRO68K……定価￥58,000▶特価￥40,000
◆CZ-224LSD The 福袋 Ver.2.0……定価￥9,980▶特価￥7,400
◆CZ-225BSD Multiword Ver.1.1……定価￥32,000▶特価￥23,000
☆ゲームソフト25% OFF OK.//（一部ソフト除く）

◆CZ-243BSD CYBERNOTE PRO68K……定価￥19,800▶特価￥15,000
◆CZ-247MSD MUSIC PRO68K（MIDI）……定価￥28,800▶特価￥20,500
◆CZ-249GSD CANVAS PRO68K……定価￥29,800▶特価￥22,000
◆CZ-251BSD Hyper word……定価￥39,800▶特価￥29,400
◆CZ-253BSD CARD PRO68K Ver.2.0……定価￥29,800▶特価￥22,700
◆CZ-257CSD Communication PRO68K Ver.2.0……定価￥19,800▶特価￥15,300
◆CZ-258BSD Teleportion PRO68K……定価￥22,800▶特価￥16,900
◆CZ-261MSD MUSIC studio PRO68K Ver.2.0……定価￥28,800▶特価￥21,200
◆CZ-263GWD Easypaint SX-68K……定価￥12,800▶特価￥9,800
◆CZ-265HSD New Print Shop Ver.2.0……定価￥20,000▶特価￥15,400
◆CZ-266BSD Press Conductor PRO68K……定価￥28,800▶特価￥22,000
◆CZ-267BSD CHART PRO68K……定価￥38,000▶特価￥29,800
◆CZ-272CWD Communication SX68K……定価￥19,800▶特価￥14,500
◆CZ-275MWD SOUND SX68K……定価￥15,800▶特価￥11,500
◆CZ-284SSD OS-9/X68000 Ver.2.4……定価￥35,800▶特価￥25,600
◆CZ-285LSD C-Compiler PRO68K Ver.2.1……定価￥44,800▶特価￥32,500
◆CZ-286BSD BUSINESS PRO68K Popular……定価￥28,000▶特価￥20,500
◆CZ-287SS SX-WINDOW Ver.2.0……定価￥12,800▶特価￥9,800

ACCESS

for X68000

V70アクセラレータの魅力を探る

V70 アクセラレータ

高速処理を実現

V70(μPD70632)は、日本電気㈱が開発した32ビットCMOSマイクロプロセッサである。このマイクロプロセッサは、数々の高度な特徴を備えており、いわゆるマイクロプロセッサのカテゴリーとしては、CISCに属する。V70アクセラレータは、このCISCチップを利用したハードウェアとしては最高峰に位置するものである。

また、V70は、それ自身浮動小点演算機構を内蔵しており、高速演算が可能であるが、更に高速、高精度な演算を行う目的で、アドバンスト・フローティング・ポイント・プロセッサ(AFPP)が標準で搭載されている。このAFPPには、右表に挙げるような特徴があり、非常に魅力的なチップなのである。

たとえばコンピュータグラフィックス等、高度な処理を要求されるシーンで、その威力を充分に発揮する。V70アクセラレータで、きみのX68000がスーパーワークステーションへと生まれ変わるのだ。

簡単に利用できる

通常アドオンCPUボードを利用する場合、そのCPUにプログラムを実行させるのもなかなかたいへんである。たとえばV70CPUにプログラムを実行させるには、まず、V70側にリセットをかけ、X68000より共有RAMの最上位アドレス部にV70側のスタートアッププログラムをロードし、リセットを解除する。V70CPUは、0FFFFFFF0Hより実行を開始する。もちろん、この後V70アクセラレータとX68000の間で適切なやりとりをして、目的とするプログラムをV70アクセラレータのローカルRAMエリアにロードし、実行して行かなければならない。

本来ならば以上のような手順をとらなければならないが、通常、ユーザはここで説明したような操作を行う必要はない。なぜならば、付属のシステムモニタ、コマンドシェルが、そのようなやりとりをすべて行ってくれるからである。

AFPPの特徴

○コプロセッサ・プロトコル内蔵
- V70対応

○高速数値演算
- システム・クロック：16MHz(最大 20MHz)
- 四則演算(倍精度)：0.9MFLOPS*(CPU：V70、20MHz動作時)
- 三角関数演算(倍精度)：6.0μsec(CPU：V70、20MHz動作時)
- ベクトル／行列演算(倍精度)：5.8MFLOS*(CPU：V70、20MHz動作時)

○IEEE754標準に準処

○豊富な命令セット
- 転送／変換命令
- 浮動小数点演算命令(四則、比較、その他)
- 制御命令
- 初等関数演算命令(三角、逆三角、指数、対数、双曲線)
- ベクトル／行列演算命令

○5種類のデータ・フォーマット
- 2進実数：32ビット、64ビット、80ビット
- 2進正数：32ビット、64ビット

○32本のデータ・レジスタ内蔵(80ビット幅)

○CPUと並行に動作可能

○68ピンPGAパッケージ

開発環境の充実

アセンブラ・リンカはもちろん、開発の強力な味方であるソースコードデバッガやシステムモニタ、さらにはフロートエミュレータ・コマンドシェルまでついている。32ビットマイクロプロセッサV70の特徴である仮想記憶、メモリプロテクション、CPUレベルでのデバッグ機能などをサポートしている。おまけにCコンパイラはというと、Human68k上のCコンパイラと互換性が高く、プログラムをほとんど修正なしで実行できてしまうのである。

アセンブラ

- 数百におよぶ命令セット、20種類におよぶアドレシングモードすべてサポート。
- コプロセッサ命令をフルサポート。
  1命令で浮動小数点演算が可能。

システムモニタ

- 仮想メモリモードを採用。
  16MByteのメモリ空間をサポート。
  大きなアプリケーションでも実行可能。
  (同時使用可能メモリ2MByteまで)。
- X68000のIOCSやHuman68kとほぼ同時のシステムコールが利用可能。

ソースコードデバッガ

- コンソールモード、リモートモード、フルスクリーンモードの3つの画面モードを持つ。
  状況に合わせたデバッグが可能。
- C言語のソースレベルでのデバッグをサポートし、C言語レベルでの式の評価、行単位、関数単位でのデバッグ可能。

フロートエミュレータ

- Human68k上の従来のアプリケーションを変更せずに、そのまま高速な浮動小数点演算が可能。

コマンドシェル

- V70用アセンブラ、コンパイラなどで記述されたV70の実行プログラムを、Human68kの実行形式プログラムを実行するのと同様の感覚で実行する環境を提供。

《オプション》Cコンパイラ

V70アクセラレータ用のC言語で開発するためのCコンパイラ。
C標準ライブラリの他、X68000本体のシステムコールを利用するための、DOSコールライブラリやIOCSコールライブラリも用意。

※製作：ボード………有限会社アクセス
ソフトウェア……株式会社ハドソン

価格
●ボードパッケージ (XVI対応)
VDTK-X68K…………¥248,000
●オプションソフト(Cコンパイラ)
VDTK-C-X68K………¥68,000

購入方法
上記商品は当面の間、通信販売とさせて頂きます。
購入ご希望の方は、住所、(社名、所属)氏名、電話番号をお知らせ下さい。
注文書をお送りいたします。

有限会社アクセス
〒101 東京都千代田区神田神保町1-64
神保町協和ビル7F
☎03(3233)0200㈹ FAX.03(3291)7019

パソコン/ワープロ通信ネットワークサービス

J&P HOT LINE

第2回　KANAさん　ID:JH194651

パソコン通信を通じてフリーソフトを駆使されるなど、X68000の魅力を最大限に引き出されている、熱心なパワーユーザー、KANAさんの登場です。今回は、使いこなしの秘術をうかがってみました。

基本データ

- ■使用機種名:CZ-612C-BK
- ■主な周辺機器:カラーイメージユニット・MIDIボード・SCSIHD モデム・MIDI音源など
- ■使用開始時期:1990年10月から
- ■好きなX68000用フリーソフト:カラオケPRO-68K（MIDIやFM音源の音楽に合わせて歌詞を表示）
- ■HOTLINEのおすすめコーナー:SIG（CZ-CLUB・SHARP-HOTLINE・DOGACGA NET）

■X68000購入の理由は?

68000を使った純国産パソコンである事。その基本性能の高さから、常にクリエイティブな発想をさせてくれるマシンだった事。先進的なデザイン、ソフトハウスやユーザー達の作る他に類を見ないソフトウェア、そして何よりもX1の思想を受け継いだパソコンである事が選んだ理由だと思います。(他にそういったマシンは、当時(現在も)国産パソコンになかった)

■主にどんなふうな用途で使われていますか?

パソコン通信 ／ ホームベースであるJ&P HOTLINEをよく利用しています。他に関西の草の根BBSや他大手ネットなどにアクセスしています。

ゲーム ／ シューティングゲームやアクションゲームが好みで、各ソフトハウスから発売された優秀なソフトウェアを今までにたくさん買いました。

音楽鑑賞 ／ MIDI音源はMT−32を持っているのですが、RCシステムと呼ばれるフリーソフトウェアのMIDI演奏ドライバは素晴しいもので、各種のフォーマットからなるMIDIデータ(98やMacなども含めて)を演奏させて聞いています。

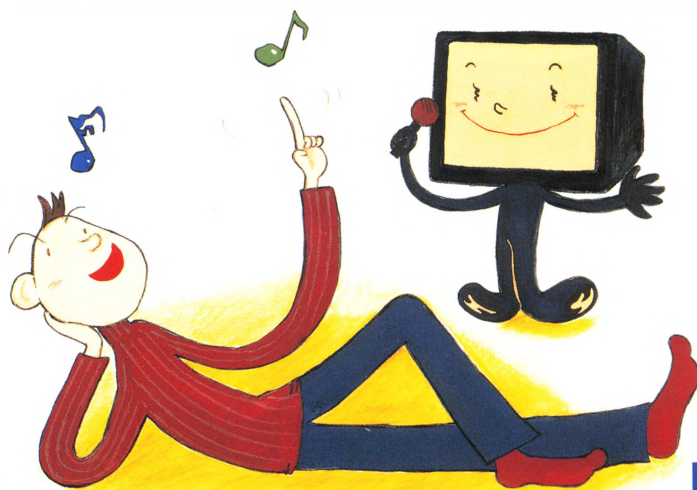
その他、CG鑑賞・プログラミング・DTVと多彩に使っています。

■X68000を使っての満足感、よいところ、楽しい部分は?

X68000は、ユーザー数は少なく、近くに持っている人間も少ない。市販ソフトも多くはなく、時には他機種の人気作品をプレイできない時もあり、周辺機器や言語ソフトなども選択肢が少ない。でも、こんな不満が、すべてパソコン通信で解決されます。市販ソフトの使いにくい点や、ない機能を加えたフリーソフトなど、普通に考えれば信じられないような事が可能になる!

X68000にはパソコン通信が不可欠なのです!(僕の持論です(笑))

■あなたにとって、J&P HOTLINEとは?

我が家。とにかく、初めてパソコン通信を体験した場所であり、今も自分が普段いる場所になっているから。

J&P HOT LINEへのご入会はスタータキットで。

お求めは、下記のお店へ。又は現金書留にて、¥3,000+¥90(消費税3%)=¥3,090を事務局までお送り下さい。
すぐにスタータキットをお送りします。

お問い合わせは
〒556 大阪市浪速区日本橋西1-6-5 上新電機株式会社
J&P HOTLINE事務局宛　TEL.(06)632-2521

スタータキットのお求めはJ&P各店でどうぞ。

店名	住所	電話
渋谷店	東京都渋谷区道玄坂2−28−4	☎(03)3496-4141
町田店	東京都町田市森野1−39−16	☎(0427)23-1313
八王子店	東京都八王子市旭町1-1八王子そごう7F	☎(0426)26-4141
立川店	東京都立川市幸町4−39−1	☎(0425)36-4141
三鷹店	東京都三鷹市野崎1−20−17	☎(0422)31-6251
横浜店	横浜市西区北幸2-9-5横浜HSビル1F	☎(045)313-6711
本厚木店	神奈川県厚木市中町3−4−4	☎(0462)25-5151
津田沼店	千葉県習志野市津田沼1-11-2	☎(0474)72-5211
焼津インター店	静岡県焼津市越後島385	☎(054)626-3311
富山店	富山市掛尾町300	☎(0764)22-5033
金沢店	金沢市入江2−63	☎(0762)91-1130
寺地店	金沢市寺地2−3	☎(0762)47-2524
大須店	名古屋市中区大須4−2−48	☎(052)262-1141
テクノランド	大阪市浪速区日本橋5−6−7	☎(06)634-1211
メディアランド	大阪市浪速区日本橋5−8−26	☎(06)634-1511
コスモランド	大阪市浪速区難波中2−1−17	☎(06)634-3111
U. S. LAND	大阪市浪速区日本橋4−9−15	☎(06)634-1411
ビジネスランド	大阪市北区梅田1-1-3大阪駅前第3ビルB2	☎(06)348-1881
高槻店	高槻市高槻町11−16	☎(0726)85-1212
くずは店	枚方市楠葉花園町15−2	☎(0720)56-8181
千里中央店	豊中市新千里東町1-3 SENCHU PAL 2番街4F	☎(06)834-4141
摂津富田店	高槻市大畑町24−10	☎(0726)93-7521
寝屋川店	寝屋川市緑町4−20	☎(0720)34-1166
枚方ハイハス店	枚方市田口3−41−7	☎(0720)48-1211
藤井寺店	藤井寺市岡2−1−33	☎(0729)38-2111
岸和田店	岸和田市土生町2451−3	☎(0724)37-1021
さんのみや1はん館	神戸市中央区八幡通3−2−16	☎(078)231-2111
西宮店	西宮市河原町5−11	☎(0798)71-1171
伊丹店	伊丹市昆陽池1−63	☎(0727)77-5101
姫路店	姫路市東延末1-1住友生命姫路南ビル1F	☎(0792)22-1221
京都寺町店	京都市下京区寺町通仏光寺下ル恵比須之町549	☎(075)341-4411
京都近鉄店	京都市下京区烏丸通七条下ル東塩小路町702	☎(075)341-5769
和歌山店	和歌山市元寺町4−4	☎(0734)28-1441
和歌山南店	和歌山市中島368	☎(0734)25-1414
学園前店	奈良市学園北1−8−10	☎(0742)49-1411
奈良1はん館	奈良市三条町478−1	☎(0742)27-1111
新大宮店	奈良市法華寺町83−5	☎(0742)35-2611
郡山インター店	大和郡山市横田693−1	☎(07435)9-2221
田原本店	奈良県磯城郡田原本町千代574-1	☎(07443)3-4041
熊本店	熊本市手取本町4−12	☎(096)359-7800

T1002179030603 雑誌 02179-3